Universidad Complutense de Madrid

Facultad de Informática

Proyecto Portal FDI

Una herramienta para la mejora de la gestión de la Facultad de Informática

Proyecto de Sistemas Informáticos

Madrid, Junio 2015

Autor:

Patrick Sanzo Curran

Director:

Iván Martínez Ortiz

SE AUTORIZA A LA UNIVERSIDAD COMPLUTENSE A DIFUNDIR Y UTILIZAR CON FINES ACADÉMICOS, NO COMERCIALES Y MENCIONANDO EXPRESAMENTE A SUS AUTORES, TANTO LA PROPIA MEMORIA, COMO EL CÓDIGO, LOS CONTENIDO AUDIOVISUALES INCLUSO SI INCLUYEN IMÁGENES DE LOS AUTORES, LA DOCUMENTACIÓN Y/O EL PROTOTIPO DESARROLLADO.

Autor:

Patrick Sanzo Curran

People think that computer science is the art of geniuses but the actual reality is the opposite, just many people doing things that build on eachother, like a wall of mini stones.

Donald Knuth

Agradecimientos

En primer lugar, quería agradecer a mis padres el apoyo económico y de todo tipo que me han brindado a lo largo de estos años. Sin su ayuda no habría sido posible nada de esto. En segundo lugar, a la universidad pública española, que, poco a poco, va desapareciendo. También quería agradecer por su música a David Robert Jones, Commandeur des Arts et des Lettres. Por último, quiero agradecer al tutor del proyecto todo su esfuerzo y colaboración en este proyecto.

Table of Contents

Índice de ilustraciones

Índice de abreviaturas

API	Aplication Programming Interface
BBDD	Base de datos
CSRF	Cross-Site Request Forgery
CSV	Comma-separated values
CRUD	Create-read-delete-update
GPL	General Public License
JPA	Java Persistence API
JSP	JavaServer Pages
MVC	Modelo-Vista-Controlador
REST	Representational State Transfer
STS	Spring Tool Suite
JSON	Javascript Object Notation
UCM	Universidad Complutense de Madrid
URI	Uniform resource identifier
URL	Uniform Resource Locator
RSS	Rich Site Summary
IDE	Integrated development environment
HTML	HyperText Markup Language
AJAX	Asynchronous JavaScript and XML
HSQLDB	Hyper SQL Database
JDBC	Java database connectivity

ORM	Object-relational mapping
JEE	Java Platform, Enterprise Edition
XML	Extensible Markup Language
LDAP	Lightweight Directory Access Protocol
SOAP	Simple Object Access protocol

Resumen

En los últimos años el uso y acceso a la web ha cambiado enormemente, en particular, se han diversificado los tipos de dispositivos mediante el uso de los teléfonos inteligentes y las tabletas dándose la situación en la que estos nuevos dispositivo año a año se están convirtiendo en el dispositivo preferido (o al menos más utilizado) para el acceso a internet. A raíz de este cambio en el paradigma acceso a internet, han surgido diversas metodologías y tecnologías que mejoran la experiencia de usuario en estos nuevos dispositivos y que tienen en cuenta las peculiaridades de las redes de datos móviles. Una de estas iniciativas es la metodología conocida como responsive web design que tiene por mejorar la experiencia de usuario facilitando que la estructura y diseño de una web se adapta de manera adecuada al dispositivo del usuario.

En este sentido, la estructura tanto el portal interno como la web pública de nuestra Facultad sufre de limitaciones en este aspecto. Asimismo, la estructura y arquitectura del portal de la Facultad sufre también de cierta rigidez que limita un desarrollo evolutivo y modular de la misma.

El objetivo de este trabajo de Sistemas Informáticos ha sido el desarrollo de un nuevo prototipo de portal para la Facultad de Informática abordando estas dos limitaciones: un diseño web adaptable a múltiples dispositivos y una arquitectura flexible; y también proporcionando nuevas funcionalidades funcionalidades no disponibles en el portal actual, como son la integración con redes sociales y nuevas funcionalidades como la gestión y reserva de espacios de la Facultad y la gestión de solicitudes de tutorías.

Palabras clave

Herramienta de gestión, gestión, facultad de informática, avisos, reservas, integración social, twitter, responsive design.

Abstract

Over the last few years, Internet has evolved rapidly with the introduction of smartphones, tablets and portable devices in general. In fact, nowadays these portable devices are becoming the preferred device (or at least the most used) for Internet access. In response to this change, many methodologies and technologies have emerged to facilitate and improve the user experience using these portable devices, taking into account the specific restrictions of the mobile networks. One of these emergent concepts is responsive web design, which aims to improve the end user experience providing a web design that easily adapts its structure to the characteristics of each device.

In this regard, the School of Computer Science has limited support to these devices, both for the intranet portal and the public website. Moreover, the internal structure and the architecture of the these portals are rigidly constructed, presenting multiple limitations that hinder an evolvable and modular development.

The main goal of this project has been the development of a new prototype for the internal portal of the School of Computer Science that addresses these two limitations: a responsive web design and more flexible architecture. In addition, this work also provides new functionalities that are not implemented in the current version of the portal, such as the management and reservation of the School's spaces (e.g. meeting rooms) and the management of requests for tutoring appointments.

Keywords

Management tool, management, school of computer science, updates, reservations, social integration, twitter, responsive design.

1 Introducción

1.1 Problema

Los dispositivos y el modelo de acceso a internet ha cambiado enormemente en los últimos años, en particular, en la actualidad y la perspectiva del futuro indica que los teléfonos móviles y las tabletas desbancan los ordenadores tradicionales en el acceso a internet [1] [2]. A raíz de este cambio en las costumbres en el consume de recursos web, han surgido diversas iniciativas y tecnologías que mejoran la experiencia de usuario en estos nuevos dispositivos y que tienen en cuenta las peculiaridades de las redes de datos móviles. Una de estas iniciativas es la metodología conocida como *responsive web design* [3] [4]que tiene por mejorar la experiencia de usuario facilitando que la estructura y diseño de una web se adapta de manera adecuada al dispositivo del usuario. La versión actual de la web de la Facultad de Informática y del portal interno es, con respecto a la tecnologías y tendencias actuales, bastante rígida y poco flexible.

La web pública de la Facultad (accesible desde http://informatica.ucm.es) está gestionada a través del Gestor de Contenidos Web de la Universidad Complutense de Madrid [5]. Además de esta web pública, la Facultad de Informática dispone de un Portal Interno utilizado por los alumnos, profesores y personal de administración y servicios de nuestra Facultad que permite automatizar ciertas tareas y procedimientos específicos de nuestra Facultad, por ejemplo, la publicación de avisos en la web pública, gestión de taquillas, etc. Este portal es una aplicación desarrollada internamente en la Facultad y, pese al gran abanico de funcionalidades que ofrece, tiene una estructura que no permite una fácil extensión para añadir nuevas funcionalidades.

1.2 Solución: Portal FDI

Como resultado de estas limitaciones, el objetivo de este proyecto de Sistemas Informáticas ha sido el desarrollo de un prototipo de un nuevo portal para la Facultad de Informática que abordara las dos limitaciones principales anteriormente identificadas: facilitar el uso de dispositivos móviles y crear una arquitectura base y flexible para añadir nuevas funcionalidades al portal.

Asimismo, debido a la limitación temporal de este trabajo, en vez de recrear todas las funcionales que actualmente se encuentran implementadas, este proyecto reemplaza una de las funcionalidades que más visibilidad tiene para los alumnos y profesores de esta

Facultad, la gestión de avisos y notificaciones incluyendo la integración con redes sociales y la creación de otros módulos como la gestión y reserva automática de espacios y la solicitud y gestión de tutorías, funcionalidades demandadas por los profesores y alumnos respectivamente.

1.3 Estructura del trabajo

A continuación se describe la estructura de los capítulos restantes de la presente memoria ofreciendo un breve resumen acerca de los contenidos que se describen en los mismos:

- **Objetivos y Requisitos.** En este apartado se describen los objetivos específicos de este proyecto, así como los requisitos técnicos y funcionales del mismo.

- **Descripción de la Herramienta.** Este apartado describe detalladamente la funcionalidad (desde el punto de vista del usuario) que ofrece la herramienta desarrollada durante este trabajo.

- **Implementación y diario de trabajo.** Este apartado describe los detalles técnicos de la herramienta desarrollada, las herramientas y tecnologías usadas, así como la planificación y modelo de desarrollo utilizado.

- **Conclusiones y trabajo futuro. En este apartado se resumen las contribuciones y se las conclusiones extraídas durante el desarrollo de este trabajo,** se abordan las posibles ampliaciones que se podrían realizar sobre la aplicación en un futuro, como parte de otro proyecto de Sistemas Informáticos, por ejemplo. También se presenta un resumen de las conclusiones obtenidas al terminar el proyecto.

2 Objetivos y Requisitos

Este capítulo presenta los objetivos principales y los requisitos funcionales de la aplicación desarrollada durante este proyecto.

2.1 Objetivos

El objetivo general de este trabajo consisten en crear una aplicación que permita gestionar, de manera sencilla y eficaz, diversas tareas de de la vida académica de la facultad tales como avisos o reservas de recursos de la Facultad.

2.1.1 Estructurar el portal de manera modular

Se quiere **estructurar el portal** de manera modular, de modo que sea posible incluir y ampliar los contenidos y las funcionalidades de forma más sencilla y abordable en posteriores trabajos de Sistemas Informáticos o trabajos de Fin de Grado. Este objetivo requiere requiere particularmente una decisión cuidadosa de la arquitectura y tecnologías a utilizar en el proyecto.

2.1.2 Integración con redes sociales y feeds de noticias

El consumo de la información y de los contenidos multimedia a cambiado en la actualidad, es decir, el buscador no es el único punto de inicio en la navegación de los usuarios, sino que el uso de redes sociales o los feeds de noticias. En este sentido, la Facultad de informática ha aumentado su presencia en las redes sociales durante el año 2014, en particular a través de la red social Twitter. Para poder publicar en redes sociales como Twitter, se requiere el uso de una cuenta específica dentro de la red social que permita identificar al usuario que realiza la publicación, dándose la necesidad (en algunos casos) de tener que compartir las credenciales de acceso a la red social si múltiples personas pueden publicar en la misma.

Por tanto, este trabajo plantea como objetivo una **mayor integración con las redes “sociales”**, como Twitter, y así aprovecharse de los beneficios que ofrecen estas plataformas, tales como: incremento de interacción entre estudiantes, profesores y demás miembros de la Facultad. Un beneficio adicional obtenido con la integración con las redes sociales es el poder integrar las noticias de la Facultad en el flujo de información que consumen los “usuarios” de la Facultad. Finalmente, esta integración debe permitir que los usuarios de la aplicación puedan generar contenidos para la red

social sin necesidad de compartir las credenciales de acceso.

2.1.3 Automatización de tareas comunes

La gestión de avisos y noticias del portal de la Facultad, es una de las funcionalidades que se utiliza más asiduamente. La funcionalidad actual es limitada, por lo que se mejorará la funcionalidad ofrecida actualmente. Asimismo, existen tareas que son gestionadas manualmente y que, por tanto, son susceptibles de automatizarlas para facilitarlas y agilizar.as. Algunas de estas actividades son la reserva de espacios comunes (Sala de Reuniones, Sala Auxiliar de reuniones, Sala de Grados, etc.) o la solicitud de tutoría de un alumno y su posterior confirmación del profesor.

2.1.4 Adaptar web a nuevos tipos de dispositivos

Como ya se ha mencionado, el tipo de dispositivos que se utiliza para acceder a internet y en particular para utilizar el portal de la Facultad es muy variado, por lo que se hace necesario que el diseño web del nuevo portal tenga en cuenta estas necesidades.

Asimismo, el buscador Google ha comenzado recientemente (abril de 2015) a comenzado a penalizar en sus búsquedas a aquellos sitios web que no cumplan unos criterios mínimos de usuabilidad del sitio web, para los dispositivos móviles [6].

2.2 Requisitos

En cualquier aplicación de cierta envergadura, es necesaria una documentación que establezca qué se quiere hacer, por qué y cómo se plantea hacerlo. De esta necesidad surge la **toma de requisitos** como documento de referencia acerca del funcionamiento del proyecto.

Para este proyecto, no se ha usado una toma de requisitos clásica, sino que se ha optado por usar las llamadas “u*ser stories*” para elicitar las necesidades de la aplicación. Una “User story” (historia de usuario) es una definición de alto nivel de un requisito funcional, en la que se especifica el usuario (o tipo de usuario), la acción o acciones que realiza y el el resultado esperado de esas acciones. Es un requisito más informal que un “caso de uso”, y suele ser descrito por el cliente [7] [8].

Cabe destacar que a veces, las historias de usuario, pese a tener una descripción simple y somera, su implementación puede llevar una gran carga de trabajo. Este tipo de historias de usuario se denominan “*epic stories”*. Estas *epic stories* se tienen que desglosar en otras *user stories* con un alcance más restringido y finalmente estas *user*

stories se descompondrán en tareas específicas de desarrollo.

Cabe destacar que este modelo de desarrollo basado en *epic stories*, *user stories y tasks* es adecuado para la metodología de trabajo que se ha utilizado a lo largo del proyecto (ver capítulo 4.3), de modo que el tutor del trabajo ha venido jugando el papel de cliente o *product owner* [9], por lo que no hace falta tanto detalle como en el escenario de no tener a uno de los usuarios finales de la aplicación como parte del equipo de trabajo.

2.3 User stories y epic stories

Para describir la funcionalidad de la aplicación se describirán inicialmente las *epic stories* para posteriormente dividirlas en *user stories*.

Título EPIC 1
Como **miembro de la facultad** quiero un sistema de gestión de avisos para dinamizar la información generada por la facultad.
Descripción
Simplificar gestión de avisos.

Título US 1.1
Como **miembro del personal administrativo de la facultad** quiero **crear, editar y borrar avisos.**
Descripción
Gestión de anuncios, pudiendo crear anuncios de tipo URL, enlace o HTML, añadiendo fechas para limitar su visibilidad.

Título US 1.2
Como **miembro de la facultad** quiero **ver avisos.**

Descripción

Listado de avisos, pudiendo filtrar por campos.

Título US 1.3

Como **miembro del personal administrativo de la facultad** quiero **publicar avisos en redes sociales.**

Descripción

Opción de publicar avisos en la red social Twitter.

Título US 1.4

Como **profesor o alumno de la facultad** quiero **acceder a un feed RSS** para poder recibir las noticias de la Facultad en el lector de noticias que utilizo habitualmente.

Descripción

Habilitación de url compatible con lectores RSS.

Título EPIC 2

Como **alumno o profesor de la facultad** quiero un **sistema de gestión de reservas** para evitar conflictos y aprovechar mejor los recursos de la facultad

Descripción

Simplificar gestión de reservas.

Título US 2.1

Como **miembro del personal administrativo de la facultad** quiero **crear, editar y eliminar** los **espacios** de los que pueden disponer los profesores y alumnos de la facultad.

Descripción

Opción de publicar avisos previamente creados en la red social Twitter.

Título US 2.2

Como **alumno o profesor de la facultad** quiero **crear, editar y eliminar** una **reserva** asociada a un recurso de la facultad en una fecha determinada por mí.

Descripción

Gestión de reservas sobre un calendario para ver la ocupación temporal de un recurso

Título EPIC 3

Como **alumno o profesor de la facultad** quiero un **sistema de gestión de tutorías** para registrar, simplificar y estandarizar la manera en la que se acuerdan tutorías.

Descripción

Gestión uniforme de tutorías

Título US 3.1

Como **alumno o profesor de la facultad** quiero **proponer una tutoría** a un profesor o alumno, respectivamente.

Descripción

Creación de tutorías, apareciendo la tutoría creada en el listado tanto del profesor como del alumno.

Título US 3.2

Como **alumno o profesor de la facultad** quiero **notificar al receptor de la propuesta** de tutoría para que confirme la tutoría.

Descripción

Envío de email de solicitud de tutoría al correo asociado al receptor tras crear una tutoría. Se envía adjunto un archivo ".cal" y otro ".csv" para poder importarlo a un software de gestión de calendarios.

Título US 3.3

Como **alumno o profesor de la facultad** quiero **notificar al emisor de la propuesta** de tutoría que he confirmado la tutoría.

Descripción

Envío de email de confirmación de tutoría al correo asociado al emisor tras confirmar una tutoría.

Título EPIC 4

Como **usuario** quiero un **servicio que acorte una url** para facilitar su transmisión de enlaces en avisos o tweets.

Descripción

Servicio de acortamiento de URL.

Título US 4.1

Como **alumno o profesor de la facultad** quiero **acortar una url** rápidamente para poder compartirla al instante.

Descripción

Creación de URL cortas a través de un servicio interno REST via AJAX, siendo el sufijo una cadena de caracteres que no sea fácilmente enumerable

Título	US 4.2
Como **miembro del personal administrativo de la facultad** quiero poder acceder a un listado completo de las URL. ***Descripción*** Listado de URL cortas, junto a la URL original, y el número de visitas, para usar esta información con fines estadísticos.	

Título	EPIC 5
Como **administrador de la aplicación** quiero un **sistema de gestión de usuarios.** ***Descripción*** Servico que permite gestionar usuarios	

Título	US 5.2
Como **administrador** quiero **dar de alta a un usuario** a la aplicación. ***Descripción*** Creación de usuarios a través de un formulario.	

Título	US 5.3
Como **usuario** quiero modificar mis datos. ***Descripción*** Modificación de datos a través de un formulario.	

3 Descripción de la Herramienta

Esta herramienta permite al usuario, dependiendo de sus privilegios realizar las siguientes tareas:

- **Gestionar usuarios**, pudiendo éstos conectarse a la aplicación usando simplemente su cuenta de la UCM.
- **Gestionar avisos**, pudiéndolos publicar en la red social twitter.
- **Gestionar reservas** de los recursos de los que dispone la facultad.
- **Acortar URLs** relativas a la apliación o externas.
- **Gestionar tutorías**, confirmándolas cuando sea oportuno.

3.1 Módulo de Gestión de Usuarios

Este móduloe permite crear y eliminar usuarios, además de establecer las credenciales de los mismos.

Para crear un usuario, basta con rellenar el formulario de la Ilustración 1. Hay dos restricciones a la hora de crear un usuario:

- El nombre de usuario (username) no puede existir.
- Las contraseñas introducidas deben coincidir.

Como ejemplo, puede verse en la siguiente imagen lo que ocurre si se introducen contraseñas no coincidentes.

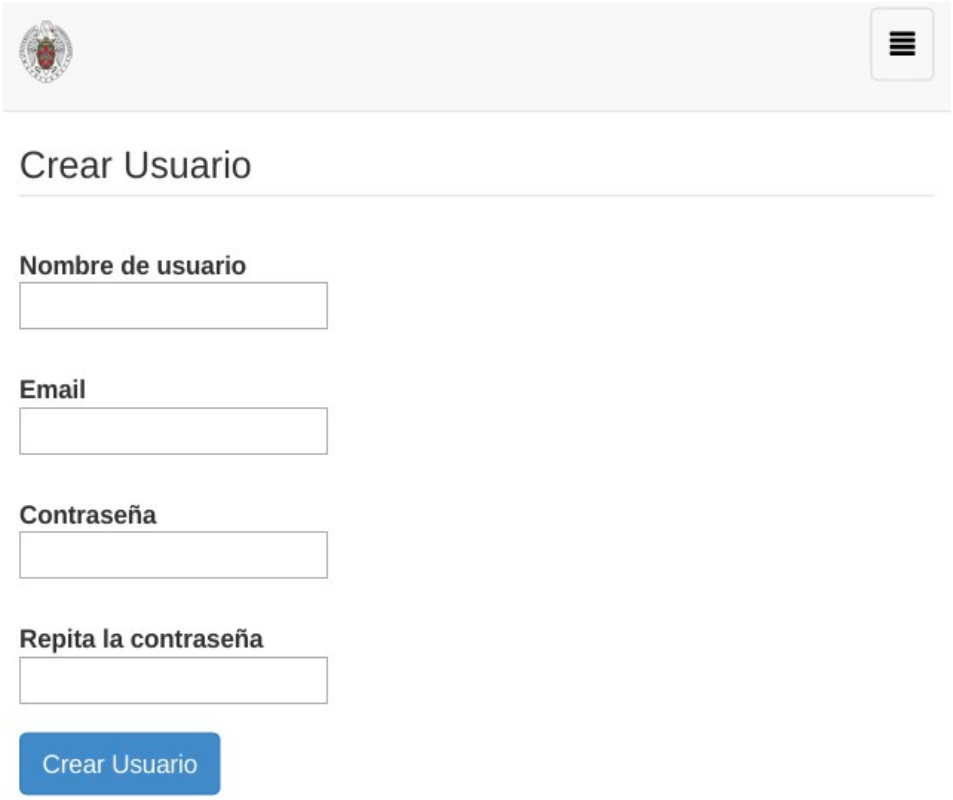

Ilustración 1: Creación de usuario

Para crear o editar un usuario, basta con rellenar el formulario (ver Ilustración 2), con dos restricciones:

- El nombre de usuario (username) no puede estar dado de alto en la base de usuarios.
- Las constraseñas introducidas deben coincidir.

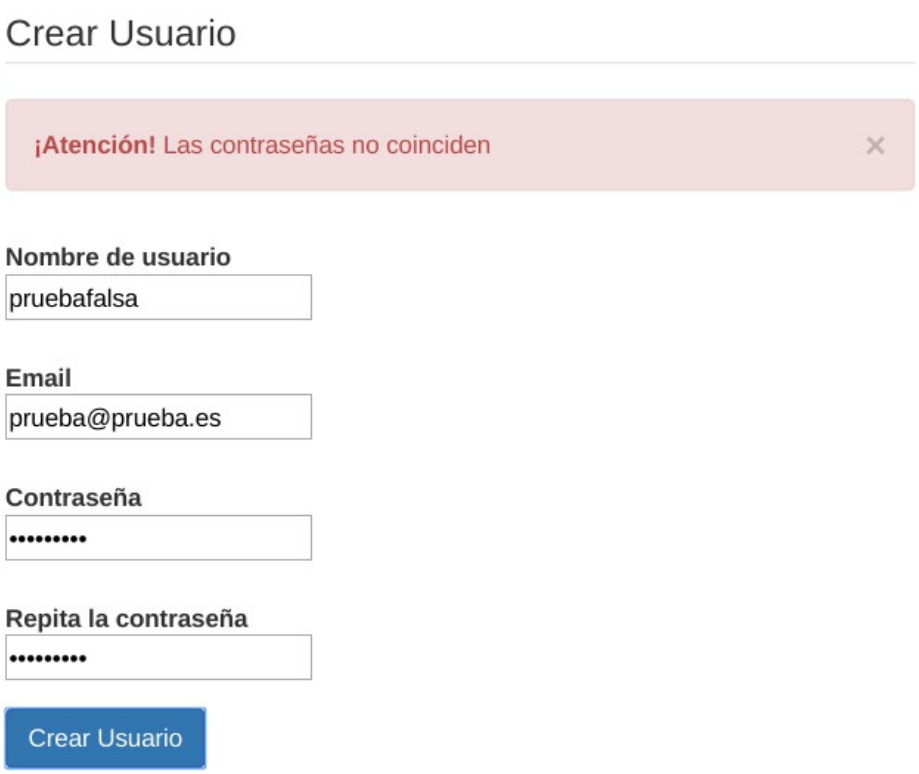

Ilustración 2: Introducción de contraseñas diferentes

Si el usuario tiene el rol de administrador, para facilitar tareas de gestión, puede listar los usuarios actuales del sistema (ver Ilustración 3).

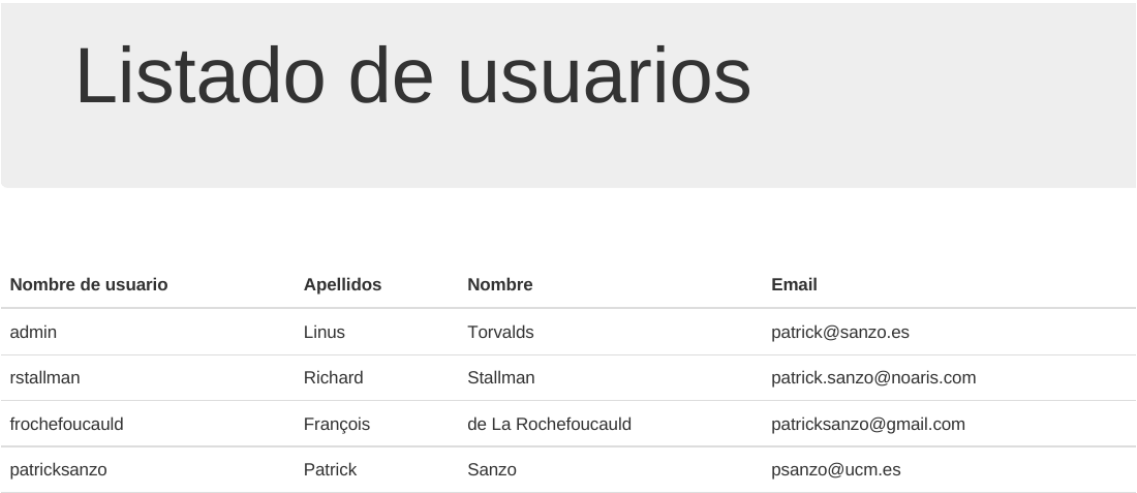

Ilustración 3: Listado de usuarios (perfil administrador)

Por último, el login del módulo de usuarios dispone de la funcionalidad “remember me” (recuérdame), que permite recordar la identidad de un usuario entre sucesivas sesiones (ver Ilustración 4).

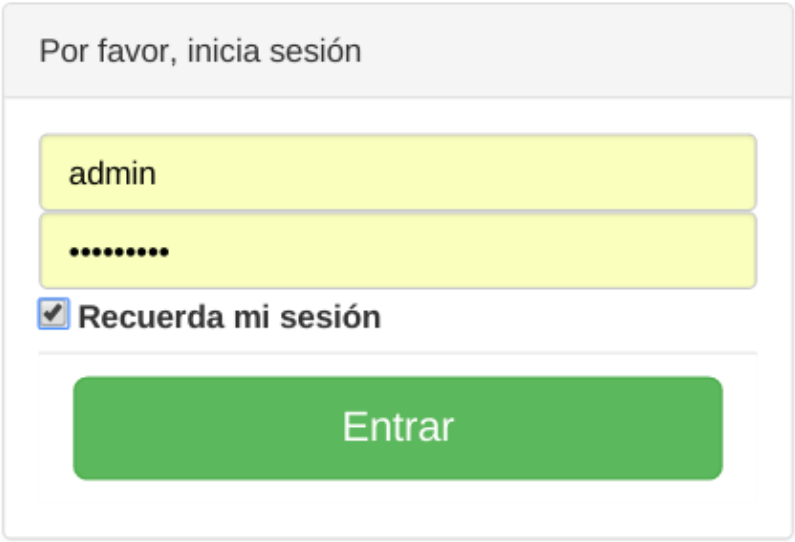

Ilustración 4: Funcionalidad remember me

3.2 Módulo de anuncios

Este módulo permite gestionar anuncios teniendo en cuenta que:

- Existen anuncios de tres tipos: URL, HTML y Adjunto.
- Todos los tipos de avisos disponen de una fecha de comienzo y de fin, para mostrarse o no dependiendo de la fecha actual.
- Además, los avisos se pueden etiquetar para poder organizar avisos por temática.

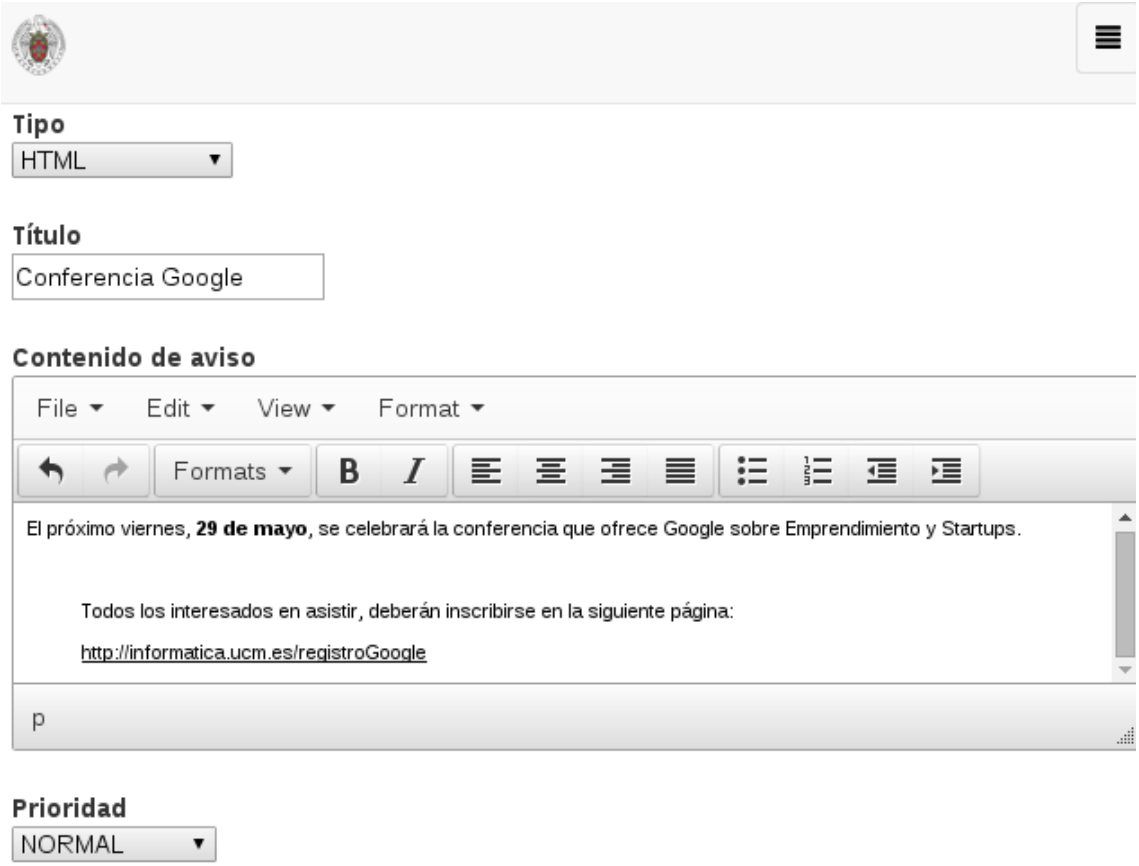

Ilustración 6: Creación de aviso de tipo HTML

Los diferentes tipos de aviso permiten:

- Los avisos de **tipo URL** (ver Ilustración 9) están pensados como una noticia corta que consiste en un título y un enlace que apunta a una página autoexplicativa.

- Los avisos **HTML** (ver Ilustración 6) están pensados para ser autocontenidos, ya que se puede insertar más información y dispone de una mayor flexibilidad de presentación. Para implementar la opción de añadir contenido formateado con HTML se ha usado una librería javascript llamada **tinymce**.

- Los avisos de tipo **Adjunto** (iIlustración 7) están pensado como un simples enlace a un documento adjunto. Para implementar el guardado de los archivos

que se quieren adjuntar se ha usado una clase-controlador que permite servir archivos cuando se solicitan a través de una URL predefinida, y una clase-servicio que permite guardar y consultar archivos adjuntados previamente.

Ilustración 7: Creación de aviso de tipo Adjunto

Un aviso puede dejarse con la fecha de comienzo de publicación o fin de publicación (o ambas) vacía, y el sistema se encargará de asignar unas valores por defecto a las fechas, como se puede comprobar en las ilustraciones 8 y 9.

Tipo	Título	Contenido del aviso	Etiqueta	Tweet	Publicado por	Fecha creación	Comienzo Publicación	Fin Publicación
	Aviso URL		defecto		admin	2015/03/05 13:30	2015/03/06 13:30	2015/03/06 13:50

Ilustración 8: La fecha vacía se rellena automáticamente

Ilustración 9: Creación de aviso de tipo URL sin fecha

Para dar mayor visibilidad a las noticias de la Facultad, se ha incorporado la opción de publicar (“tuitear”) los avisos en Twitter (ver ilustraciones 10, 11 y 12), la popular red social de microblogging.

Al pulsar el botón de tuitear (Ilustración 10) , sale una ventana modal (ver Ilustración 11)que permite editar el aviso antes de que sea enviado a Twitter.

Ilustración 10: Botón para tuitear un aviso

Ilustración 11: Creación de "tweet" en base a un anuncio.

Ilustración 12: Tweet creado a partir de anuncio, disponible en twitter.

Este módulo también expone el listado de anuncios a través de un **feed RSS** como se muestra en la Ilustración 13. Esta funcionalidad permite estar suscrito a las noticias de la facultad, recibiendo con la frecuencia de la actualización deseada los nuevos avisos, siendo avisado cuando haya un nuevo aviso.

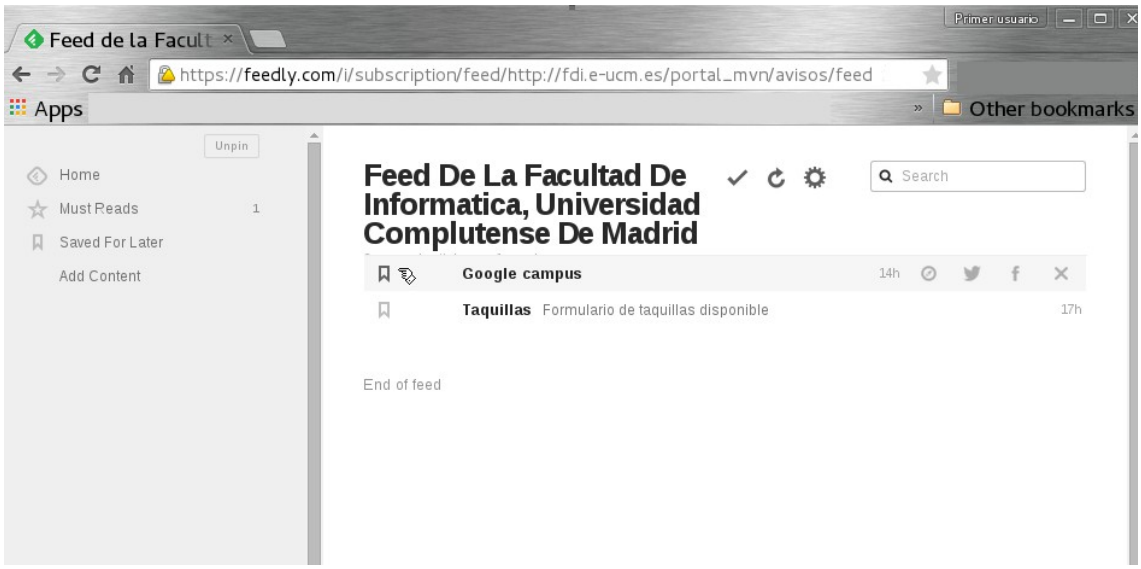

Ilustración 13: Feed Rss de avisos visto desde la aplicación Feedly

3.3 Módulo de Reserva de espacios

Este módulo facilita la reserva de espacios (o recursos) de los que dispone de la facultad. Dispone de un listado de reservas simple (ver Ilustración 14), con su correspondiente formulario de creación de reserva (ver Ilustración 15) y de un calendario que permite ver las reservas además de crear una reserva seleccionando la duración de la reserva, para luego elegir el recurso del que se quiere disponer (ver Ilustración 16).

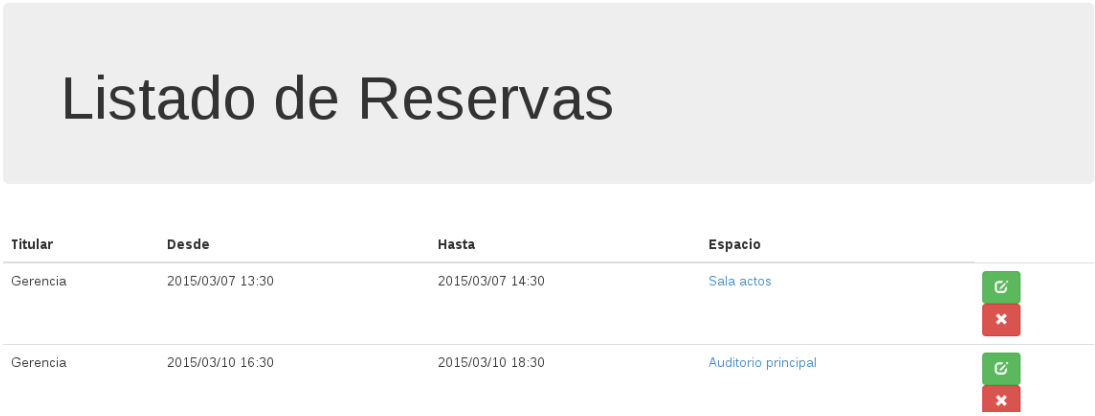

Ilustración 14: Listado simple de reservas

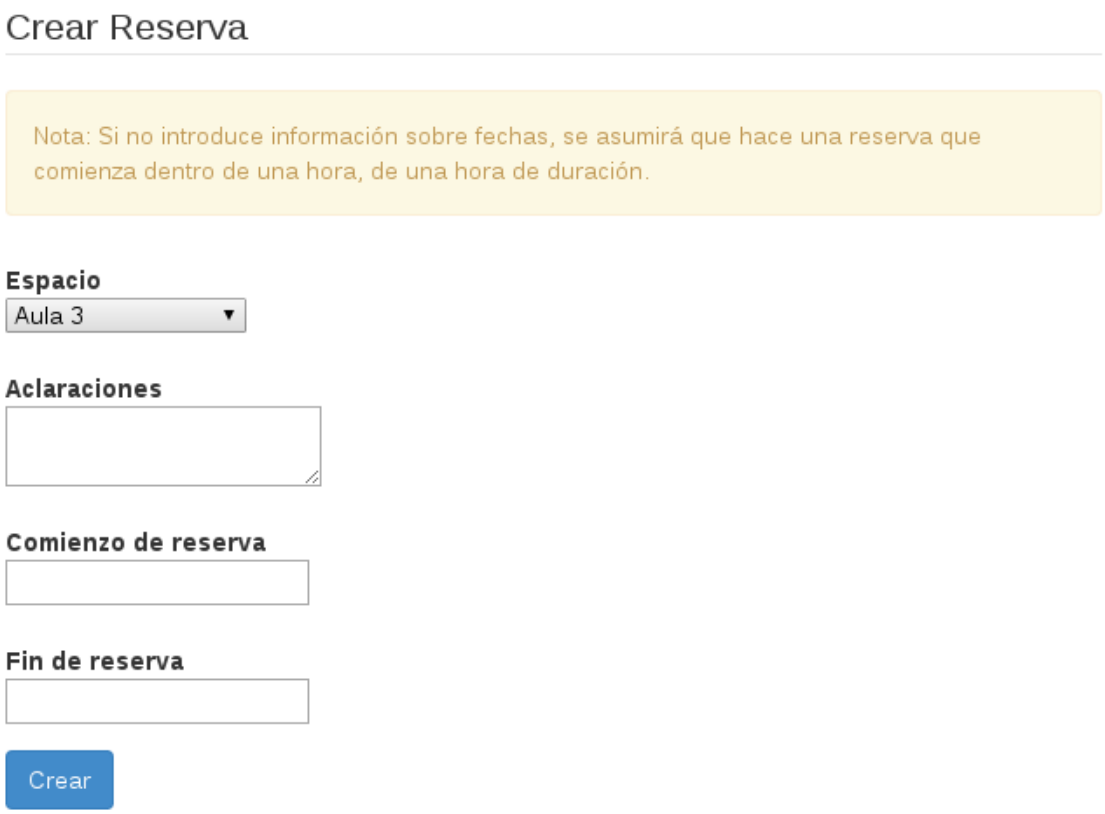

Ilustración 15: Creación de reserva desde formulario

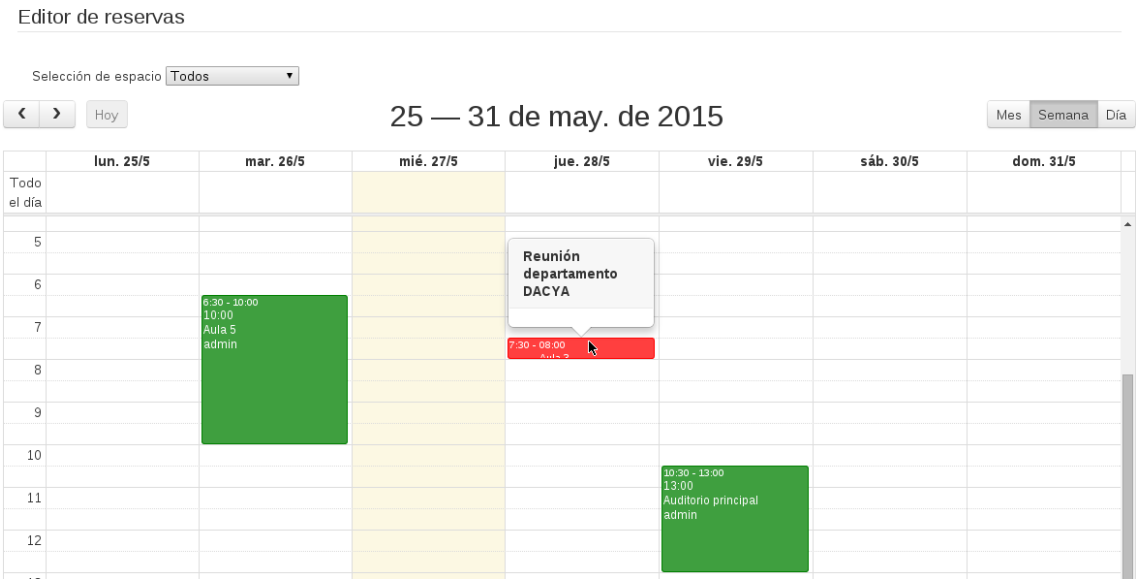

Ilustración 16: Calendario de reservas de recursos de la facultad.

La lista de los espacios se puede modificar (si se dispone de los permisos necesarios) para adecuarla a la disponibilidad de la facultad, como se ve muestra en la Ilustración 17.

Ilustración 17: Creación de espacio

Para mayor facilidad de gestión se dispone de un listado simple de espacios, como se ve en la Ilustración 18.

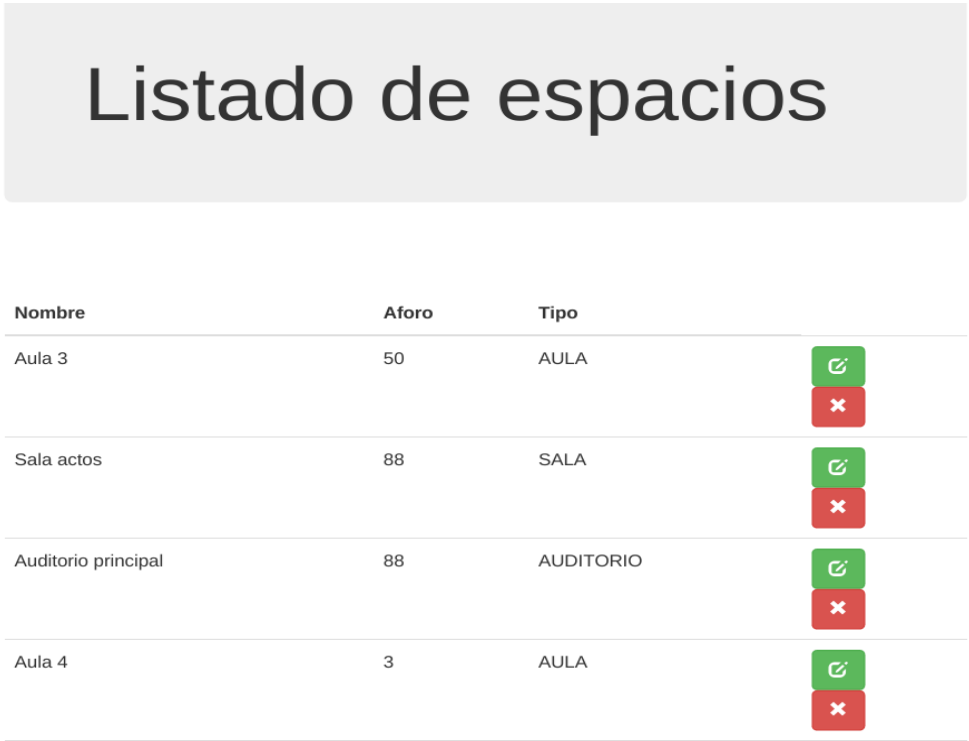

Ilustración 18: Listado simple de espacios

3.4 Módulo de enlaces cortos

Para facilitar la gestión de avisos, y permitir enlazar una url corta, se ha creado un módulo que permite generar urls de longitud corta y sencillas de recordar a partir de una url dada.

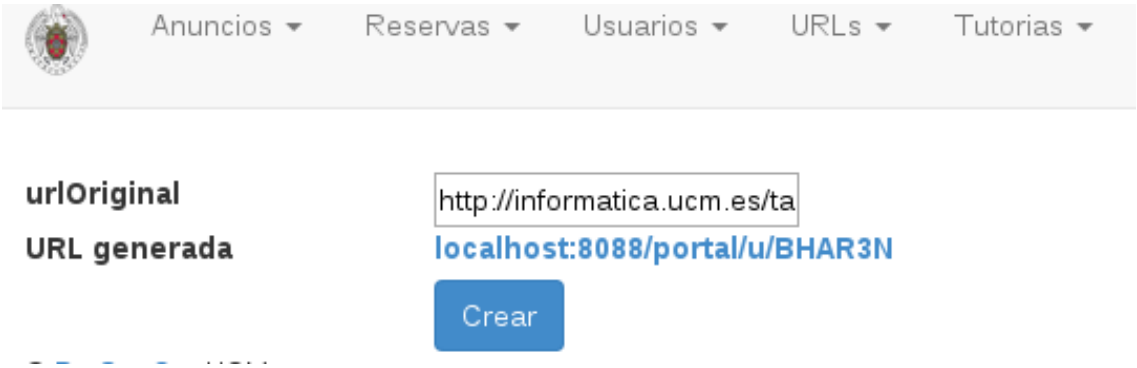

Ilustración 19: Acortador de URL de forma dinámica.

Si se accede a una URL acortada cuya URL original es interna a la aplicación (un anuncio, por ejemplo), se redirigie directamente a la URL de la aplicación. Si, por el contrario, es una URL externa, se muestra un mensaje que avisa de que el usuario va a ser redireccionado a una URL externa, y, al cabo de unos segundos, redirecciona al usuario a la página deseada.

3.5 Módulo de tutorías

Este módulo permite crear tutorías, indicando un resumen del motivo,. Se especifica también la asignatura a la que se refiere la tutoría, el profesor al que va dirigida la tutoría, y el intervalo de tiempo en que se quiere realizar (ver Ilustración 22).

Al solicitar una tutoría, el sistema enviará un email de solicitud de tutoría al destinatario de ésta con un enlace que, al pulsarse (e identificarse en caso de no estarlo), confirma la tutoría (ver Ilustración 23).

La tutoría llevará dos archivos adjuntos que la describen: uno en formato “csv” y otro en formato “cal”. De esta manera, el usuario podrá importar la descripción y la fecha de la tutoría a su software de gestión de calendarios. Alternativamente, Gmail (desde la interfaz clásica, no desde “Inbox”) ofrece la posibilidad de añadir la tutoría a Google Calendar, como se ve en la descripcrión de los eventos del mensaje en la Ilustración 20.

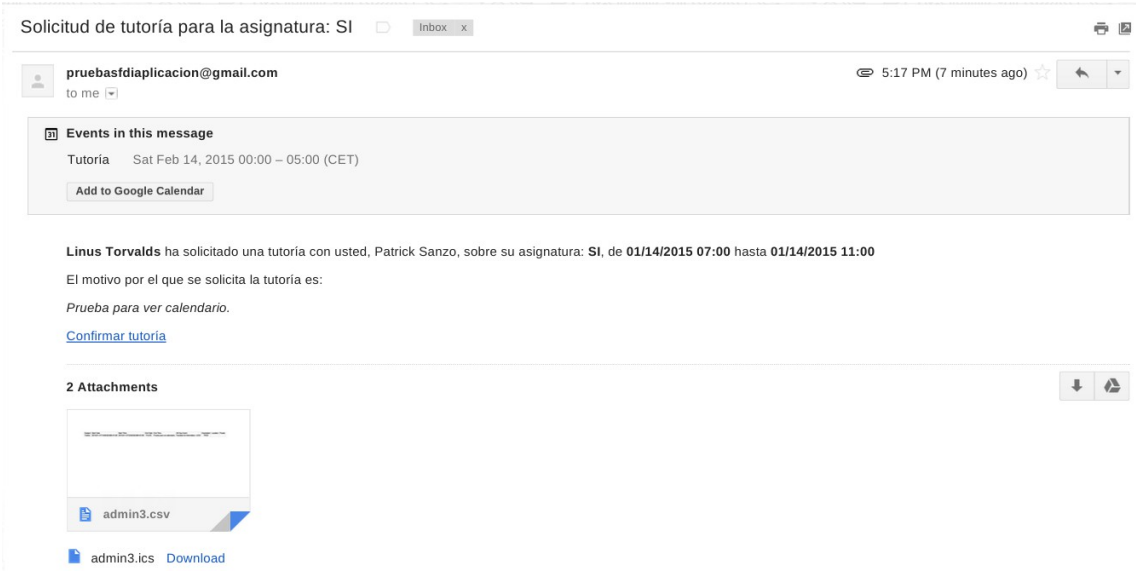

Ilustración 20: Email de solicitud de tutoría con enlace para confirmar tutoría

Si se importa la tutoría al calendario de la cuenta de Google (ver Ilustración 34), se actualizarán todos los detalles relevantes para la tutoría, tales como la hora, la descripción o el lugar, además de incluirse una referencia al correo desde el que se importa la tutoría.

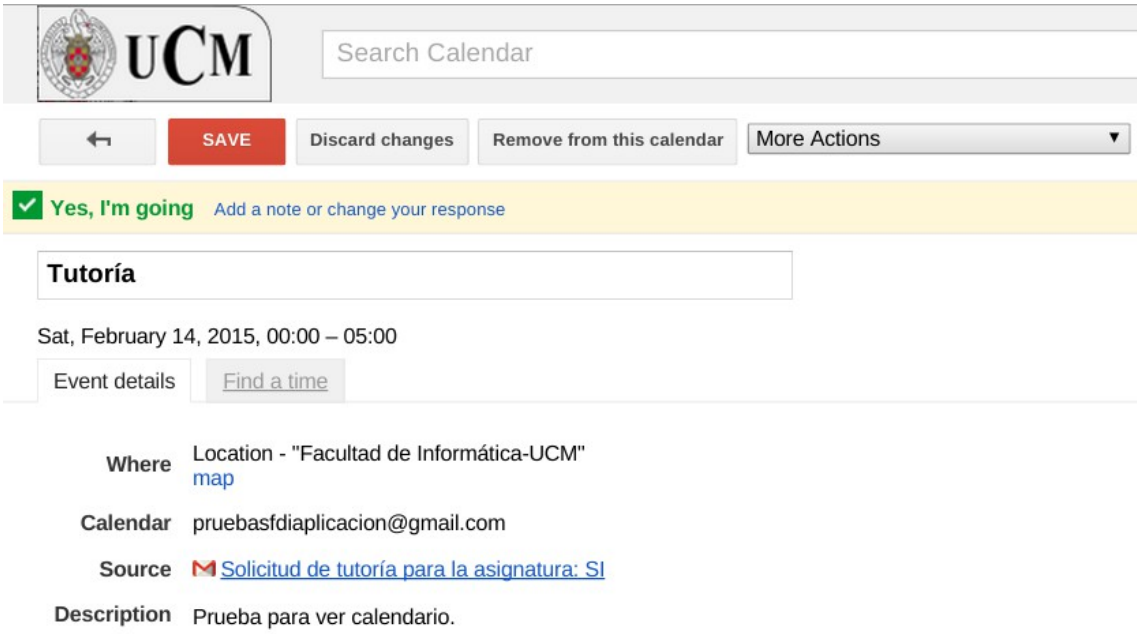

Ilustración 21: Tutoría importada al calendario de Google

Tras confirmar el destinatario la tutoría, se enviará otro email, en este caso al solicitante de la tutoría, indicándole que su tutoría se ha confirmado (ver Ilustración 24).

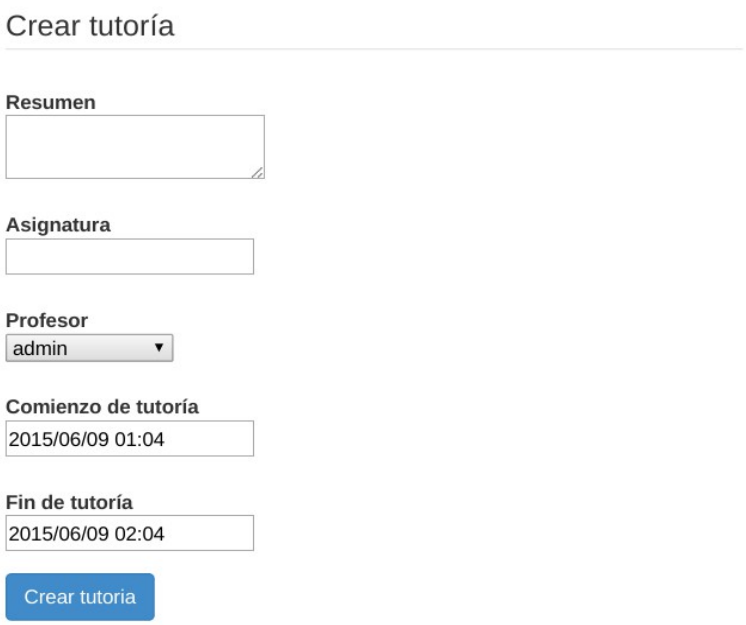

Ilustración 22: Creación de tutoría

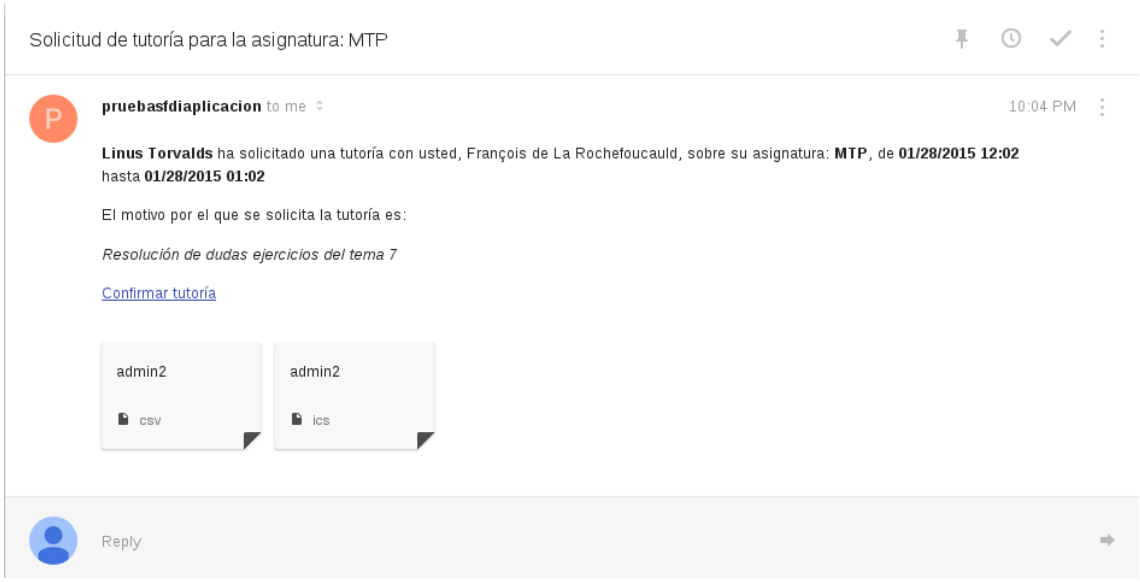

Ilustración 23: Email de solicitud de tutoría con enlace para confirmar tutoría.

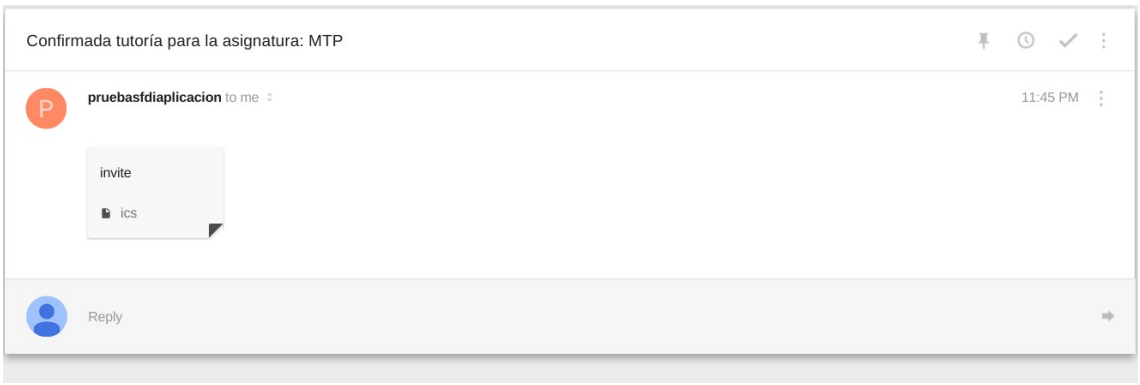

Ilustración 24: Correo de confirmación de tutoría

Para que el usuario tenga en cuenta las tutorías que tiene pendientes, se ha facilitado un listado de tutorías con el estado de las mismas. Desde este listado se pueden confirmar las tutorías pendientes (ver Ilustración 25).

Emisor	Destinatario	Asignatura	Resumen de dudas	Comienzo	Fin	¿Confirmada?
admin	frochefoucauld	MTP	dudas tema 7	2015/01/09 01:42	2015/01/09 02:42	true
admin	frochefoucauld	MTP	Revisión de examen	2015/01/09 01:46	2015/01/09 02:46	Confirmar

Ilustración 25: Listado de tutorías pendientes

4 Implementación y diario de trabajo

4.1 Tecnologías usadas

Para la realización del proyecto se han usado diferentes tecnologías y lenguajes de programación. Del lado del cliente se han utilizado el lenguaje **Javascript, HTML5** y **CSS**. Del lado del servidor se ha utilizado el lenguaje de programación **Java** y el framework **Spring** (junto a **Spring Security y Spring Social**) y finalmente **Hibernate** y **Spring Data** para la capa de persistencia.

Debido a la gran cantidad de dependencias de bibliotecas necesarias para Spring y Hibernate, ha optado por utilizar la herramienta **Maven** como herramienta para gestionar las dependencias binarias y para permitir construir la aplicación.

Java es un lenguaje orientado a objetos, basado en clases, concurrente, propiedad de Oracle. Una característica interesante de este lenguaje es que intenta permitir a los desarroladores de software ejecutar en cualquier máquina código compilado en una otra máquina, posiblemente con otra arquitectura. [10]

Javascript es un lenguaje ligero, interpretado y orientado a objetos que permite ejecutar scripts del lado del cliente. Es muy útil para generar contenido dinámicamente, y, con la biblioteca jQuery, permite manipular fácilmente la estructura de un documento HTML. [11]

4.1.1 Control de versiones

Para mantener el control de versiones se ha usado la tecnología **git**, debido a su alta flexibilidad y popularidad. Git permite crear y eliminar ramas de manera muy sencilla, lo que dinamiza la creación de código para distintas “user stories”, ya que permite, por ejemplo, comenzar a trabajar en una rama, y rápidamente desarrollar un cambio en otra rama que afecte a un requisito prioritario si las circunstancias lo requieren.

Para alojar el código del proyecto se ha usado **GitHub**, que permite usar repositorios gratuitos, siempre que dichos repositorios sean de código abierto, como este proyecto. Github también permite controlar de manera simple los requisitos y los errores que hay en la aplicación a través de **“issues”**, en los que se describe una funcionalidad requerida (o un bug en la aplicación), asignando una prioridad y/o una etiqueta y/o la versión para la que se quiere realizar (ver Ilustración 26).

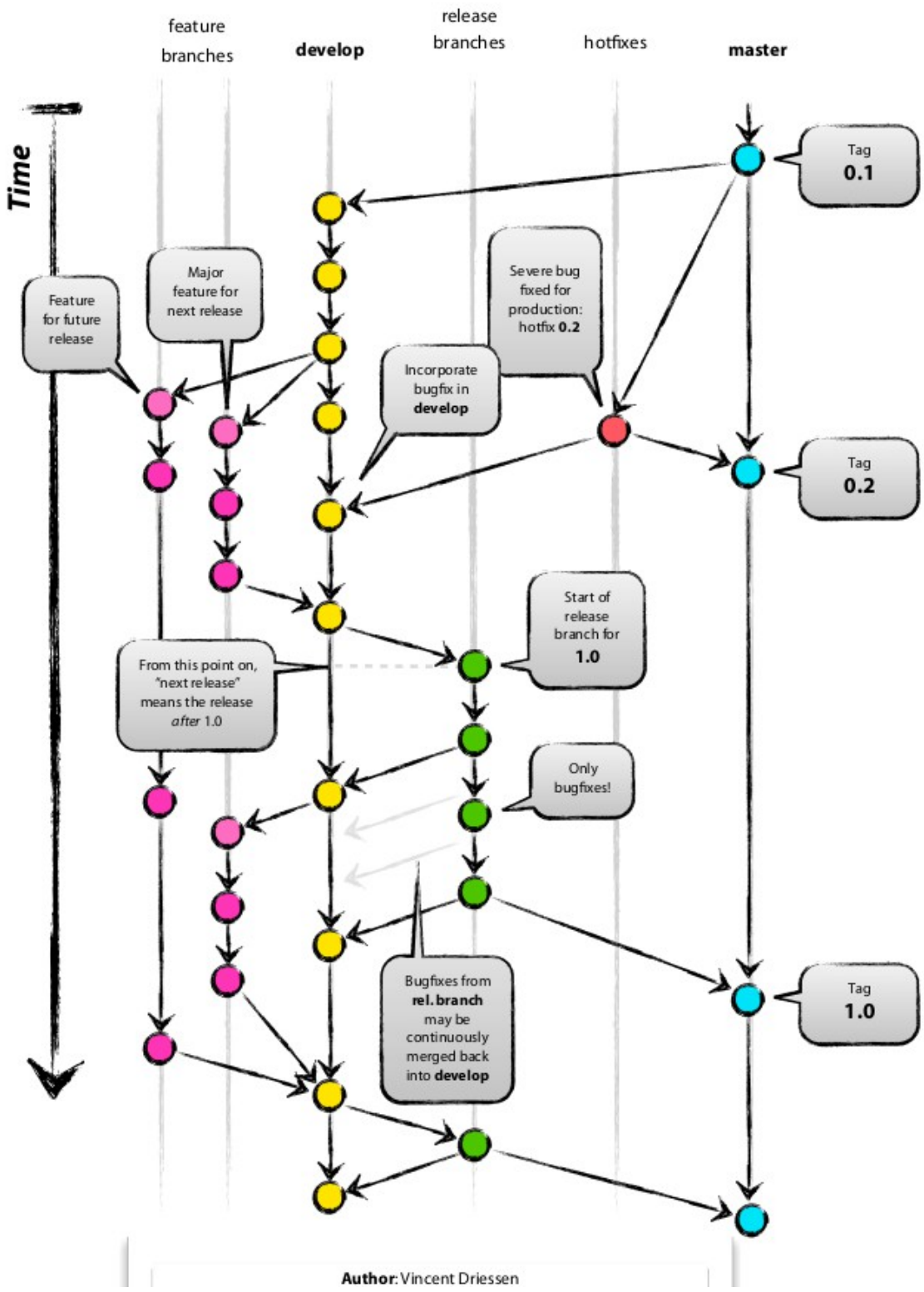

Ilustración 26: Flujo de trabajo con Git

4.1.2 Gestión de dependencias y gestión de la configuración

Para gestionar las dependencias se utiliza la herramienta de gestión de configuraciones y dependencias llamado **Maven**.

Maven permite construir proyectos abstrayendo las dependencias y conflictos que se generan entre bibliotecas. También se puede delegar en Maven la gestión de la configuración con propiedades asociadas a perfiles (desarrollo y producción, por poner un ejemplo), o propiedades que contengan valores privados (ej.: contraseñas y datos confidenciales). Todos esta información se centraliza en el archivo POM (project object model), del que depende Maven. [12]

4.1.3 Frameworks

Para facilitar la tarea de desarrollo se han usado los frameworks Spring, Hibernate y Tiles.

4.1.3.1 Spring

Spring es un **framework ligero** para construir aplicaciones basadas en el lenguaje Java. Se considera ligero porque su uso no requiere hacer muchas modificaciones sobre el código de la aplicación desarrollada para aprovechar los beneficios que el framework aporta. [13]

Aunque en un principio el proyecto Spring sólo consistía de Spring Core (núcleo), según fue evolucionando, se fueron añadiendo otros proyectos, entre los que se encuentran **Spring Data** (simplifica el acceso a bases de datos), **Spring Social** (simplifica interacciones con aplicaciones sociales), **Spring Batch** (proporciona funciones reutilizables para procesar grandes cantidades de datos), **Spring Boot** (proyecto que intenta minimizar la configuración de una aplicación, siguiendo la filosofía *Convention over configuration*), **Spring Integration** (extensión de Spring que ayuda a integrar aplicaciones hechas con Spring con sistemas externos). [10]

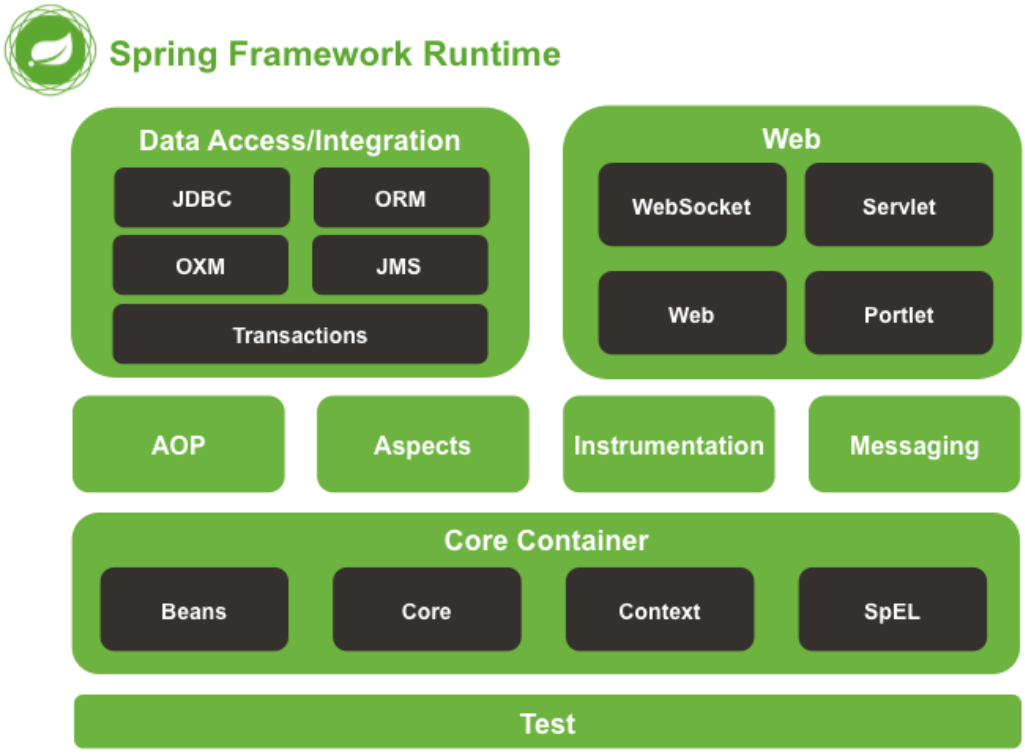

Ilustración 27: Estructura y módulos del framework Spring © Pivotal.

Las características más importantes de Spring son las siguientes [14] :

- **Inyección de dependencias e Inversión del Control**. Spring aplica el patrón de Inversión del Control mediante la inyección de dependencias. La inyección de dependencias consiste en que, en lugar de que un objeto construya los componentes de los que depende, sean inyectados a él. Esto hace que el objeto no tenga por qué tener conocimiento de los detalles del componente que se le está inyectando, ya que simplemente se delega la acción de construir el objeto en el framework (en este caso Spring). Spring facilita la construcción e inyección de dependencias, lo que hace que la aplicación esté mejor estructurada.

- **Programación orientada a aspectos**. La programación orientada a aspectos es un paradigma de programación cuyo objetivo es modularizar más el código permitiendo separar módulos cuyos fines son distintos. Se consigue añadiendo (fuera de la lógica de negocio) acciones a realizar en distintos puntos del código. Con esto se consigue que la lógica de negocio no se entremezcle con código no

esencial para ese módulo (por ejemplo registrar una acción para depurar). [15]

- **Web MVC.** Capa que surte de funcionalidades relacionadas con la web, como carga de archivos "multipart", cliente HTTP. También contiene la implementación de Spring del patrón MVC, que gira a un gestor de peticiones (DispatcherServlet) que redirecciona dichas peticiones a los controladores (anotados con @Controller), que a su vez accederán a otros mecanismos, como servicios o repositorios.

- **Acceso a datos/integración.** Proporciona módulos para usar más fácilmente métodos relacionados con el acceso a base de datos via JDBC. Además, permite integrar fácilmente el código con frameworks ORM como Hibernate.

- **Test**. Módulo que ayuda a realizar pruebas unitarias y de integración, apoyándose en frameworks para hacer pruebas como JUnit.

4.1.3.2 Spring Security

Spring Security es el módulo de la familia Spring que ofrece la capacidad de gestionar la seguridad e integrarla con Spring.

Permite configurar la seguridad de una aplicación a través de un archivo (ya sea archivo Java de configuración o un archivo XML), declarando beans propiamente configurados. Los beans se clasifican según el ámbito al que se quieren aplicar, como los de Seguridad HTTP o el AuthenticationManager, que gestiona los accesos.

Spring Security puede configurarse en el archivo de configuración para que filtre los accesos a determinados recursos siguiendo reglas simples como comprobando si un usuario es anónimo o está logueado, a la vez que permite controlar el acceso de manera complementaria a nivel de método Java con anotaciones como @PreAuthorize.

Una de las mayores ventajas de Spring Security reside en que puede integrarse con muchas tecnologías de autenticación tales como LDAP, autenticación por OpenID o autenticación básica por formulario. [16]

4.1.3.3 Spring Social

Las páginas que se consideran "redes sociales" exponen una API para que aplicaciones de terceros usen datos de los usuarios, siempre que éstos lo consientan, para que se pueda reutilizar la información compartida en dichas redes. La integración "social" se puede entender como una colaboración entre tres actores: la red social que

proporciona la API, la aplicación que la usa, y el usuario cuyos datos se están exponiendo. [17]

Es por ello que nace el proyecto **Spring Social**. Este proyecto permite usar las API de diferentes proveedores de servicios de redes sociales, como Linkedin o Twitter, de una manera más transparente al programador, simplemente indicando las claves de acceso y las operaciones a realizar.

4.1.3.4 Spring Data

Cualquier aplicación que interactúe con BBDD conlleva una cantidad considerable de código que, muy a menudo, suele variar poco de aplicación a aplicación, en lo que a operaciones CRUD y otras operaciones simples se refiere. Es por esto que surge el proyecto **Spring Data**.

Spring Data permite que, simplemente implementando una interfaz se puedan declarar métodos (según una nomenclatura establecida) que se encarguen de realizar las operaciones mencionadas en la signatura. [18]

4.1.3.5 Bootstrap

Bootstrap es un framework para la parte visual a la que accede el usuario de la aplicación. Consta de una serie de archivos CSS para poder estructurar la página de manera sencilla, y a la vez hacerlo siguiendo la tendencia actual de las páginas webs de ser “responsive”, es decir, de ser adaptables al dispositivo desde el que se está accediendo.

También comprende una serie de ficheros Javascript que dota de mayor funcionalidad a los componentes que se definen en los archivos CSS. [19]

4.1.3.6 Hibernate

Hibernate es un framework ORM (object-relational mapping) que facilita la interacción de los accesos a bases de datos, permitiendo patrones como herencia o polimorfismo sobre los objetos mapeados a la base de datos.

Otro de los beneficios de Hibernate es que permite que los accesos a bases de datos sean transparentes para el programador, sin tener en cuenta las particularidades de cada tipo de base de datos.

Además, permite optimizar el rendimiento de las consultas admitiendo diversas estrategias de búsqueda, como *lazy fetching*, técnica que permite cargar solo los objetos

requeridos para una operación.

También permite crear asociaciones directamente en los atributos de la clase Java, sin preocuparse por la creación de claves foráneas.[20] [21]

Bibliotecas

- Bibliotecas de Java
  - Rome
  - Java mail
  - Joda time
  - Jadira usertype
  - JSTL
- Bibliotecas de Javascript.
  - Tinymce
  - jQuery
  - jQuery timepicker
  - Datatables
- Otras bibliotecas
  - Apache Tiles.

4.1.4 Servidor

Se ha usado un servidor Apache Tomcat, que permite de manera eficiente gestionar la ejecución de servlets de JEE.

4.2 Arquitectura

4.2.1 Patrones de diseño

4.2.1.1 ECB

ECB (Entity-Control-Boundary) es un patrón que, como el MVC, invita a modularizar el código, asignando los roles de Entidad, Control y Frontera. El rol de Entity (entidad) lo realizan los módulos que representan información sobre el dominio (similar al modelo de MVC). El de control lo toman los módulos que, como el Controlador de MVC, se encargan de controlar el flujo de ejecución de la aplicación. Finalmente, los módulos que actúan como frontera con elementos externos al sistema o subsistema que implementa el patrón toman el rol de Boundary (frontera). [22]

4.2.1.2 MVC

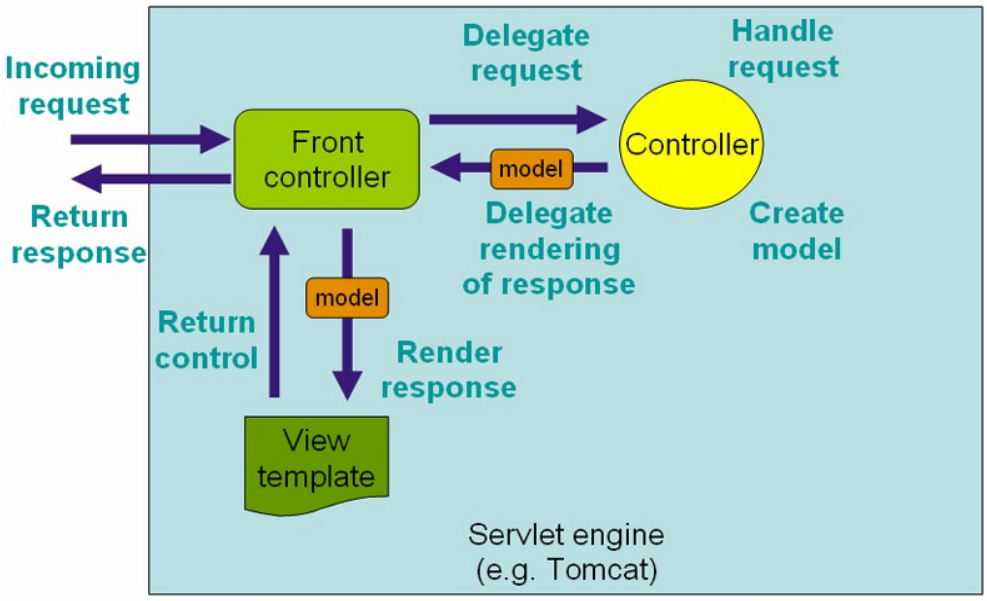

Ilustración 28: Diagrama del patrón MVC usado por Spring © Pivotal

El patrón MVC (Modelo-Vista-Controlador) establece una separación entre la vista y el modelo por una parte, y otra entre el controlador y la vista. Se entiende como **modelo** la parte que almacena información del dominio sobre el que se trabaja. La **vista** representa la interfaz del usuario con los datos de la aplicación. Finalmente, el **controlador** es el responsable de ser el intermediario entre el modelo y la vista, recogiendo la entrada del usuario, la manipula y aplica cambios al modelo. [23]

Con este patrón, se consigue que cada una de las partes desconozca el

funcionamiento de las restantes, con lo que únicamente contendrá el código necesario para desempeñar su función, sin depender de la implementación subyacente de las partes con las que se comunica.

4.2.1.3 Builder

El patrón **Builder** se usa para separar la construcción de un objeto de su representación. Permite crear diferentes representaciones de objetos con el mismo método de construcción. [24]

4.2.2 Servicio REST

Para realizar **consultas AJAX**, y para el **lector de RSS**, se han creado algunos servicios **REST**.

REST (Representational state transfer) es un tipo de arquitectura de software que permite crear una interfaz simple para acceder a determinados servicios. Las recomendaciones de este estilo permiten también una "separación de responsabilidades", con el fin de modularizar código.

Las principales propiedades de REST se pueden resumir en:

- **Interfaz uniforme** que conecte el cliente y el servidor, para que un cliente no necesite detalles de bajo nivel del servidor (ej.: almacenamiento de datos) ni un servidor sea dependiente de la implementación de la representación para el usuario de la información.

- **"Stateless"** (sin estado). Toda la información sobre la petición está contenida en la operación atómica que la contiene.

Disponde de cuatro "verbos": GET, PUT, POST y DELETE. Cada uno de ellos indica el tipo de acción que se quiere realizar. Estos verbos se acompañan de los datos correspondientes, formando así la petición REST.

Se ha optado por esta solución porque ofrece gran flexibilidad, es más sencilla de aplicar que otros servicios web de comunicación (como SOAP) y porque permitiría el acceso a los recursos de la aplicación desde una futura aplicación externa (aplicaciones móviles Android y iPhone por ejemplo).

En esta aplicación, se utliza REST para que se puedan hacer peticiones "dinámicas", a través de la técnica **AJAX**, sin tener que recargar o cambiar la página [25].

4.2.3 Persistencia

Debido a la sencillez de las operaciones de acceso a datos requeridas por la aplicación, se ha optado por usar una base de datos relacional escrita en Java, HSQLDB**.** HSQLDB es una base de datos ligera, que puede usarse tanto en memoria como en disco.

Se ha optado por usar esta base de datos directamente en memoria, para agilizar las transacciones.[26]

4.3 Diario de Trabajo

4.3.1 Metodología de trabajo

El trabajo se ha ido realizando por pequeños incrementos, realizando una reunión a la semana (aproximandamente) para la resolución de dudas por parte del alumno y la sugerencia de trabajo por parte del profesor tutor.

Se ha intentado seguir la metodología ágil en la medida de lo posible, tomando requisitos en forma de “user stories”. Esto ha sido relativamente sencillo, al no tener que depender de otras personas para aunar posiciones más que del tutor y del alumno.

Debido a su excelente integración con Spring, el IDE usado para desarrollar la aplicación es **STS** (Spring Tool Suite), una versión modificada de Eclipse. La adecuación tan buena se debe fundamentalmente a que es desarrollado por Pivotal, la empresa responsable de Spring.

4.3.2 Organización y planificación

El trabajo se ha ido avanzando completando unos hitos acordados por el tutor y el alumno. Siendo algunas tecnologías usadas en la aplicación nuevas o poco conocidas para el alumno, ha habido una fase de aprendizaje, aparte de las fases usuales de desarrollo de módulos como hitos. El desarrollo de la aplicación puede dividirse en los hitos que se comentan a continuación.

1. Conversaciones sobre el problema que se pretende resolver y cómo abordarlo.

2. Aprendizaje de Spring.

3. Aprendizaje de Hibernate.

4. Desarrollo del módulo de avisos/anuncios.
5. Desarrollo del módulo de reservas de recursos.
6. Desarrollo del módulo de usuarios.
7. Desarrollo del módulo de acortamiento de URL.
8. Desarrollo del módulo de tutorías.

4.4 Implementación de módulos

La aplicación se ha estructurado en módulos, cada uno con un servicio gestor, un repositorio (implementado con Spring Data) y una entidad que representa los datos que modela (mapeado por Hibernate), siguiendo el patrón Error: Reference source not found. Se ha aplicado también el patrón MVC, teniendo los archivos JSP representado las vistas, los controladores, y las entidades representando el modelo. Para mapear los objetos creados en los formularios, y posteriormente crear la entidad que se guardará en la base de datos, se ha utilizado el patrón Builder.

Para representar fechas en las entidades que las necesitan, se ha usado la librería Joda-Time que permite una mejor manipulación y mapeado de las fechas que la librería por defecto de Java.

Para poder mostrar fechas y seleccionarlas se ha usado la librería de jQuery/Javascript **datetimepicker**.

4.4.1 Módulo de tutorías

Este módulo se compone de un controlador que mapea las peticiones, ***TutoriasController*** y las procesa con un servicio de gestión de tutorías, ***Tutorias***, con las operaciones CRUD. Dicho servicio accede a un repositorio, ***TutoriaRepository***, gestionado por Spring Data.

Para mapear el objeto recogido por el formulario, se utiliza la clase ***TutoriaBuilder***. A partir de esta clase, se construye el objeto de la clase ***Tutoria*** que Hibernate se encargará de mapear a BBDD.

Para la realización de este módulo se ha creado una clase auxiliar que se comporta como un servicio de Emails, con métodos que permiten enviar emails preconfigurados. Hay dos tipos de emails preconfigurados:

- **Email de solicitud de tutoría.** Usa los atributos emisor, destinatario de tipo usuario, y los datos de carácter personal. Provee un enlace a través del que el usuario puede confirmar la tutoría.

- **Email de confirmación de tutoría.** Al acceder un usuario a la url de confirmación de una tutoría, y comprobarse que el usuario que confirma la solicitud de tutoría está logueado y que el contralador verifique que es el usuario apropiado para confirmar la tutoría, se llamará al servicio de emails, para que envíe al emisor un email con la confirmación de tutoría por parte del destinatario.

Para facilitar el envío de correos electrónicos se ha usado el paquete de Java JavaMail API (javax.mail).

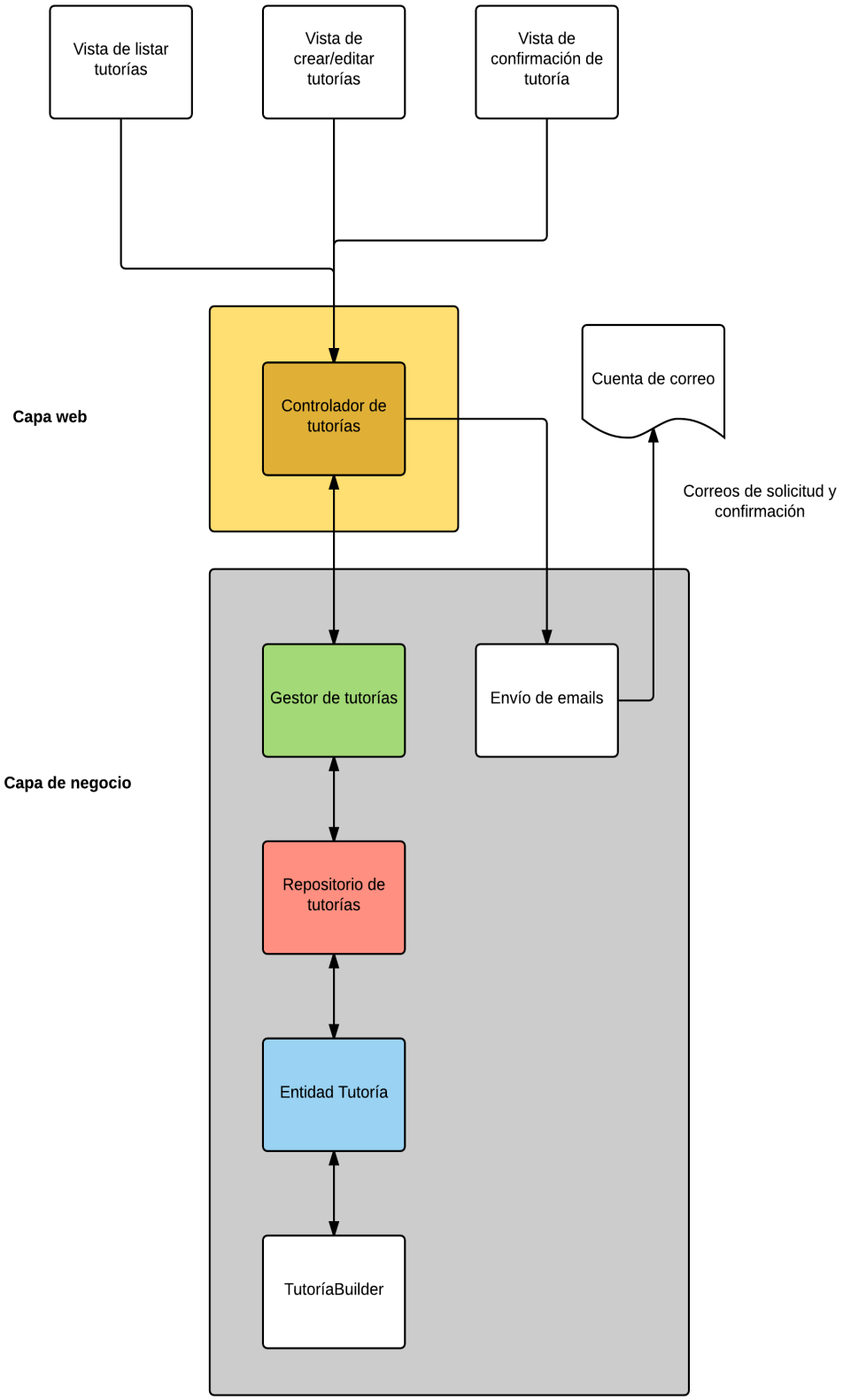

Ilustración 29: Arquitectura del módulo de tutorías

4.4.2 Módulo de anuncios/avisos

Este módulo se compone de un controlador que mapea las peticiones, ***AvisosController*** y las procesa con un servicio de gestión de avisos, ***Avisos***, con las operaciones CRUD básicas, añadiendo algunas que se han considerado útiles, como la búsqueda de avisos por etiqueta. Dicho servicio accede a un repositorio, ***AvisoRepository***, gestionado por Spring Data.

Para mapear el objeto recogido por el formulario, se utiliza la clase ***AvisoBuilder***. A partir de esta clase, se construye el objeto de la clase ***Aviso*** que Hibernate se encargará de mapear a BBDD.

Para el almacenamiento de archivos adjuntos, se dispone de un gestor de archivos, ***StorageManager***, que almacena y sirve los archivos que se adjuntan a un aviso, a partir de una ruta (relativa al proyecto) configurada en el archivo de configuración "portal-context.xml". El servicio de gestión de avisos lo llama para guardar archivos, y el controlador de almacenamiento, ***StorageController***, lo llama para servirlos. StorageManager llama a su vez a un repositorio, gestionado por Spring Data, que se encarga de almacenar los archivos cuyo guardado solicita el servicio.

Para implementar la funcionalidad de exponer un feed RSS, se ha creado una clase ***CustomRssViewer***, que se apoya en la biblioteca **rometools** para crear el feed.

Debido a la naturaleza pública de un feed RSS, se ha tenido que crear una excepción en la configuración de Spring Security, para que sea accesible a un usuario no registrado.

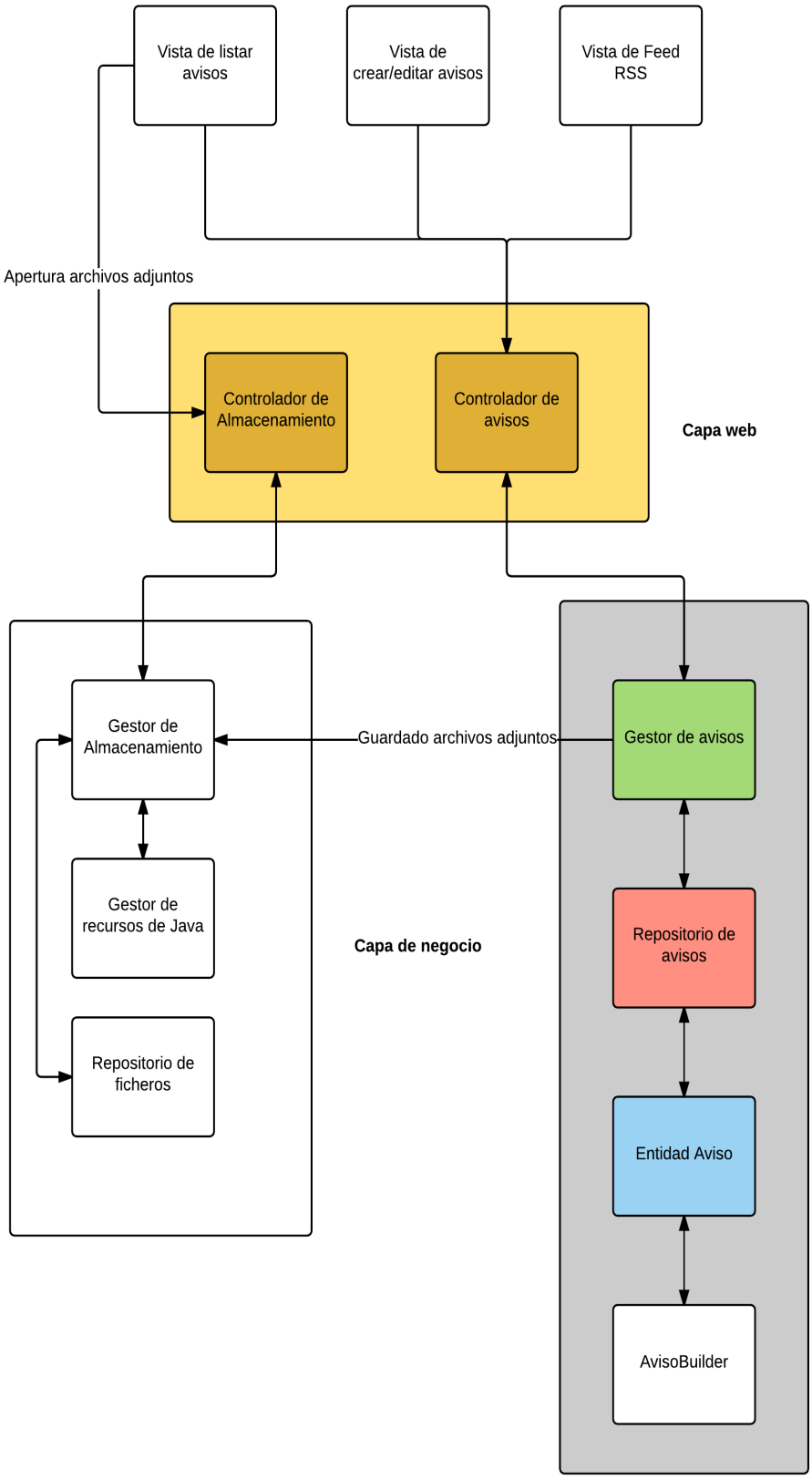

Ilustración 30: Arquitectura del módulo de avisos / anuncios

4.4.3 Módulo de acortador de URL

Este módulo se compone de un controlador que mapea las peticiones, ***AcortadorURLController*** y las procesa con un servicio de gestión de acortamiento de URL, ***URLredireccioens***, con las operaciones CRUD básicas. Dicho servicio accede a un repositorio, ***URLredireccionRepository***, gestionado por Spring Data.

Para mapear el objeto recogido por el formulario, se utiliza la clase ***URLredireccionBuilder***. A partir de esta clase, se construye el objeto de la clase ***URLredireccion*** que Hibernate se encargará de mapear a BBDD.

Al hacer una petición para acortar una URL acortada, se pide al servicio de acortamiento de URL la creación de un objeto de clase ***URLredireccion***. A partir de este objeto, se obtiene un sufijo que se añade a una raíz común (configurada en el archivo de constantes ***Constants***) a las URL acortadas.

El sufijo se genera con una función simple de Hashing y una codificación base32. Primero se aplica el Hash al id que tiene el objeto URLredireccion en la base de datos, con el algoritmo de Knuth (descrito en [27]). El resultado anterior se codifica en base32. El algoritmo utilizado se puede encontrar en [28]

Las URL se acortan a través de un servicio REST llamado desde el navegador, utilizando la técnica conocida como AJAX.

Del lado del cliente, la petición AJAX se realiza a través de la función de jQuery creada para tal propósito, **ajax**. Esta función se invoca con el tipo de contenido que se quiere enviar (en este caso JSON), el objeto que se quiere enviar y el método de envío (en este caso POST). Para que la petición no sea rechazada por el servidor, debe añadirse también la información sobre CSRF.

El mapeo de la petición REST se lleva a cabo por el método ***nuevoAcortadorREST*** del controlador. Dicho método acepta como parámetro un objeto de tipo ***URLredireccionBuilder*** *(*como el que se obtiene del formulario de creación de URL acortada*)* y devuelve un objeto de tipo URLredireccion. Gracias a las anotaciones de Spring @RequestBody y @ResponseBody y a la inclusión en el fichero “pom.xml” de la dependencia a la biblioteca **Jackson** de Fasterxml, la conversión desde/hasta JSON del lado del servidor se realiza de manera transparente al programador, lo que evita código repetitivo y propenso a fallar.

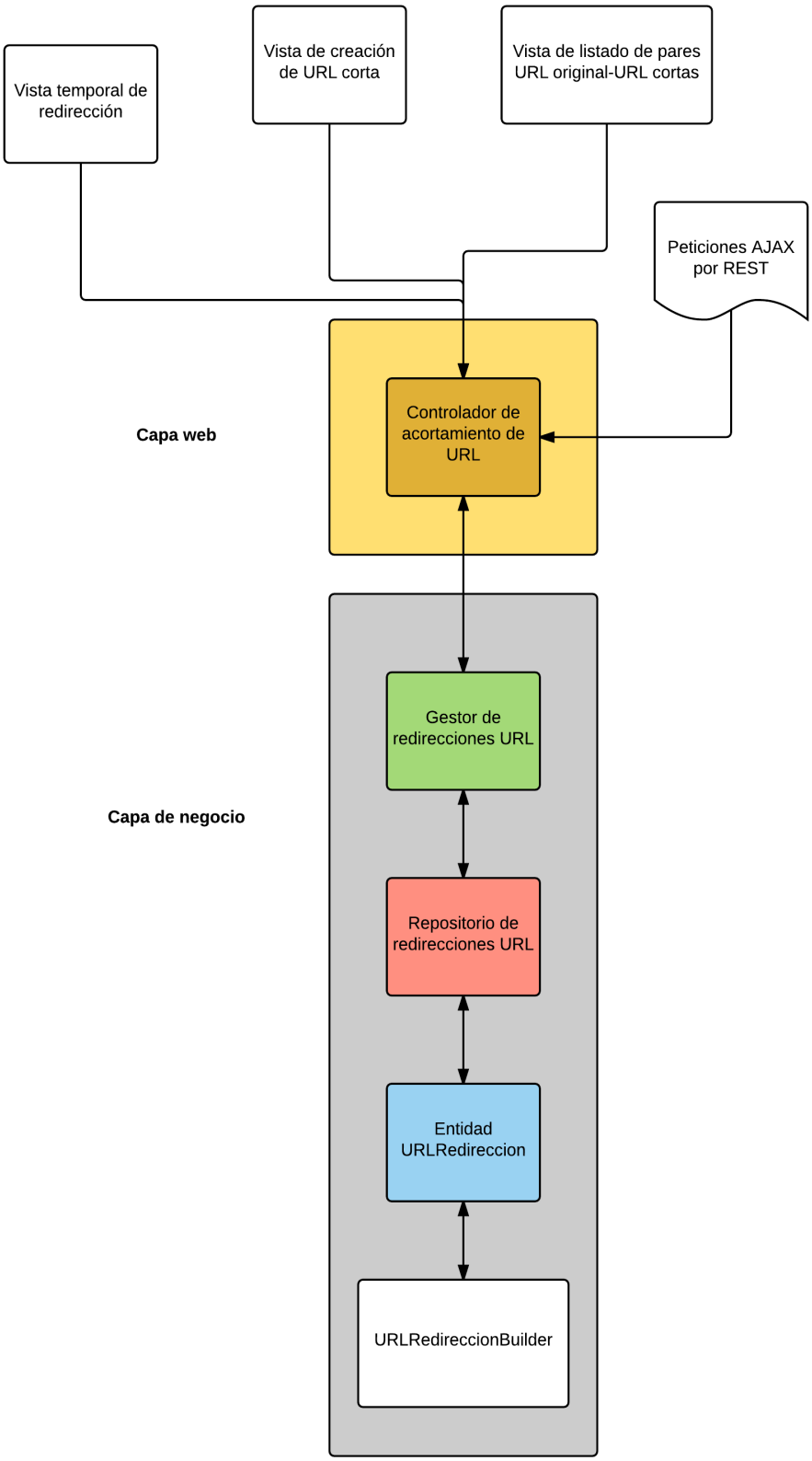

Ilustración 31: Arquitectura del módulo acortador de URLs

4.4.4 Módulo de reservas

Este módulo contiene dos submódulos: un submódulo para gestionar reservas, accesible para todos los usuarios, y otro de gestión de espacios, accesible solo desde el perfil administrativo.

El módulo de reservas se compone de un controlador que mapea las peticiones, ***ReservasController*** y las procesa con un servicio de gestión de reservas, ***Reservas***, con las operaciones CRUD básicas. Dicho servicio accede a un repositorio, ***ReservaRepository***, gestionado por Spring Data.

Para mapear el objeto recogido por el formulario de creación/edición de reservas, se utiliza la clase ***ReservaBuilder***. A partir de esta clase, se construye el objeto de la clase ***Reserva*** que Hibernate se encargará de mapear a BBDD.

El módulo de espacios se compone de un controlador que mapea las peticiones, ***EspaciosController*** y las procesa con un servicio de gestión de espacios, ***Espacios***, con las operaciones CRUD básicas. Dicho servicio accede a un repositorio, ***EspacioRepository***, gestionado por Spring Data.

Para mapear el objeto recogido por el formulario de creación/edición de reservas, se utiliza la clase ***Espacio***. Esta misma clase es la que Hibernate se encargará de mapear a BBDD.

El objeto ***ReservaBuilder*** tiene un atributo indicando el identificador del espacio asociado a la reserva. Al construir el objeto Reserva por parte de ReservaBuilder, se obtiene el objeto de tipo Espacio (obtenido mediante el servicio de espacios) que se establecerá como atributo de la reserva.

Para mostrar el calendario de reservas registradas en el sistema, se ha utilizado la biblioteca de Javascript **fullcalendar**, que permite mostrar un calendario dinámico y personalizable.

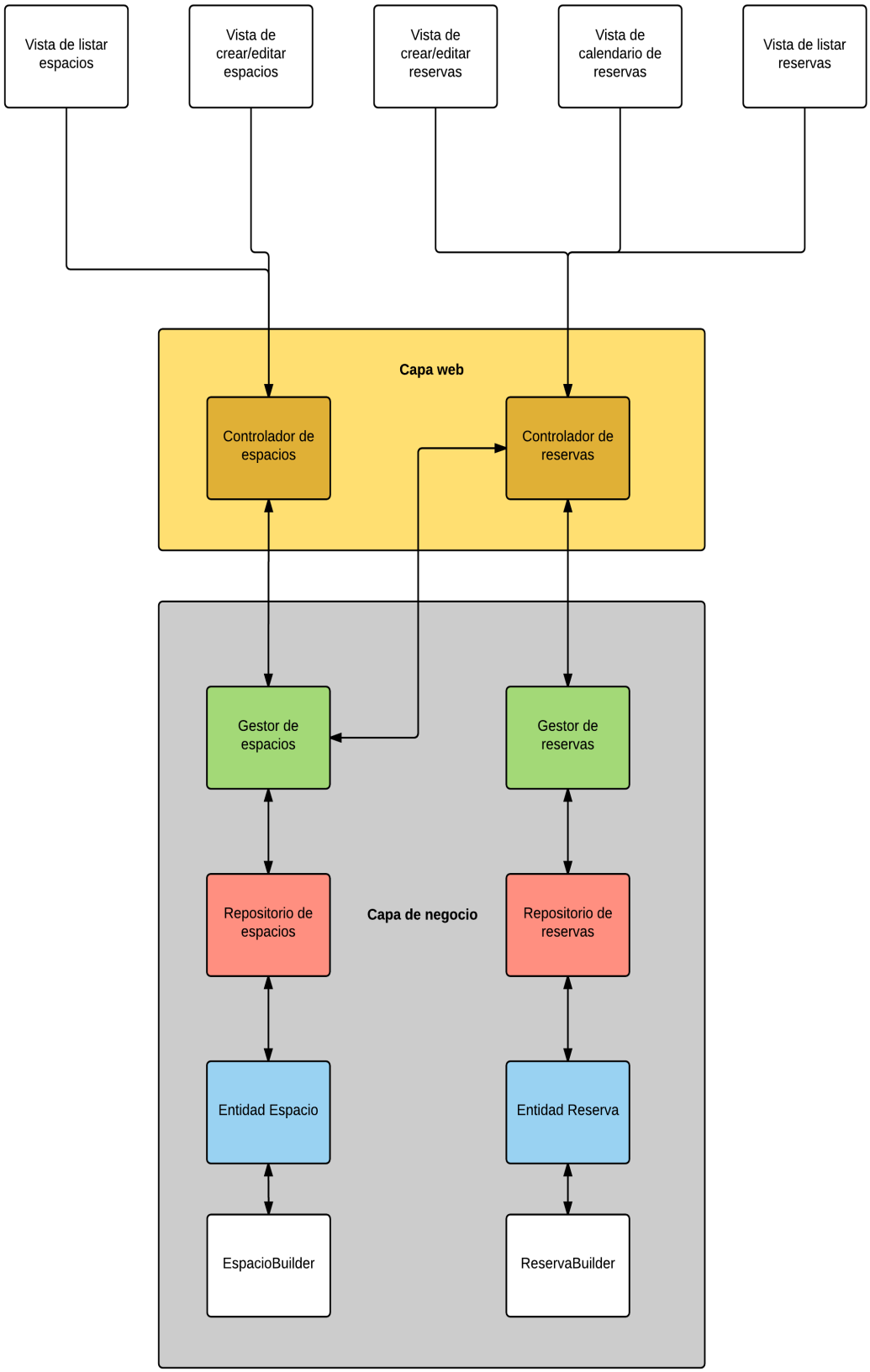

Ilustración 32: Arquitectura del módulo de reservas

4.4.5 Módulo de gestión de usuarios

Este módulo se compone de un controlador que mapea las peticiones, ***UsersController*** y las procesa con un servicio de gestión de usuarios, ***UsersManager***, con las operaciones CRUD básicas. Dicho servicio accede a un repositorio, ***UserRepository***, gestionado por Spring Data.

Para mapear el objeto recogido por el formulario, se utiliza la clase ***URLredireccionBuilder***. A partir de esta clase, se construye el objeto de la clase ***URLredireccion*** que Hibernate se encargará de mapear a BBDD.

Este módulo se apoya en una clase-entidad User que debe implementar las interfaces ***UserDetails*** y ***CredentialsContainer*** de **Spring Security Core**. La implementación mencionada permite delegar la autenticación de usuarios en Spring Security. Además, el servicio de gestión de usuarios, ***UsersManager***, tiene que implementar la clase UserDetailsService, que fuerza a sobreescribir el método ***loadUserByUsername***, que devuelve un objeto usuario en base a un nombre de usuario.

El último paso para gestionar usuarios a través de **Spring Security**, es crear una clase, que se ocupe de encriptar la contraseña.

Para encriptar las contraseñas, se ha creado una clase ***PBKDF2PasswordEncoder***, que, como sugiere el nombre, codifica la contraseña con el algoritmo **PBKDF2** (Password-Based Key Derivation Function 2), obtenido en [29].

Del lado del cliente, se hace una comprobación (usando jQuery) que compara las contraseñas coinciden,dado que es un dato esencial para el registro.

Por último, se ha incluido la opción de “remember me” en la configuración de Spring Security, pudiendo variar el lapso de tiempo en el que la cookie que recuerda al usuario es válida.

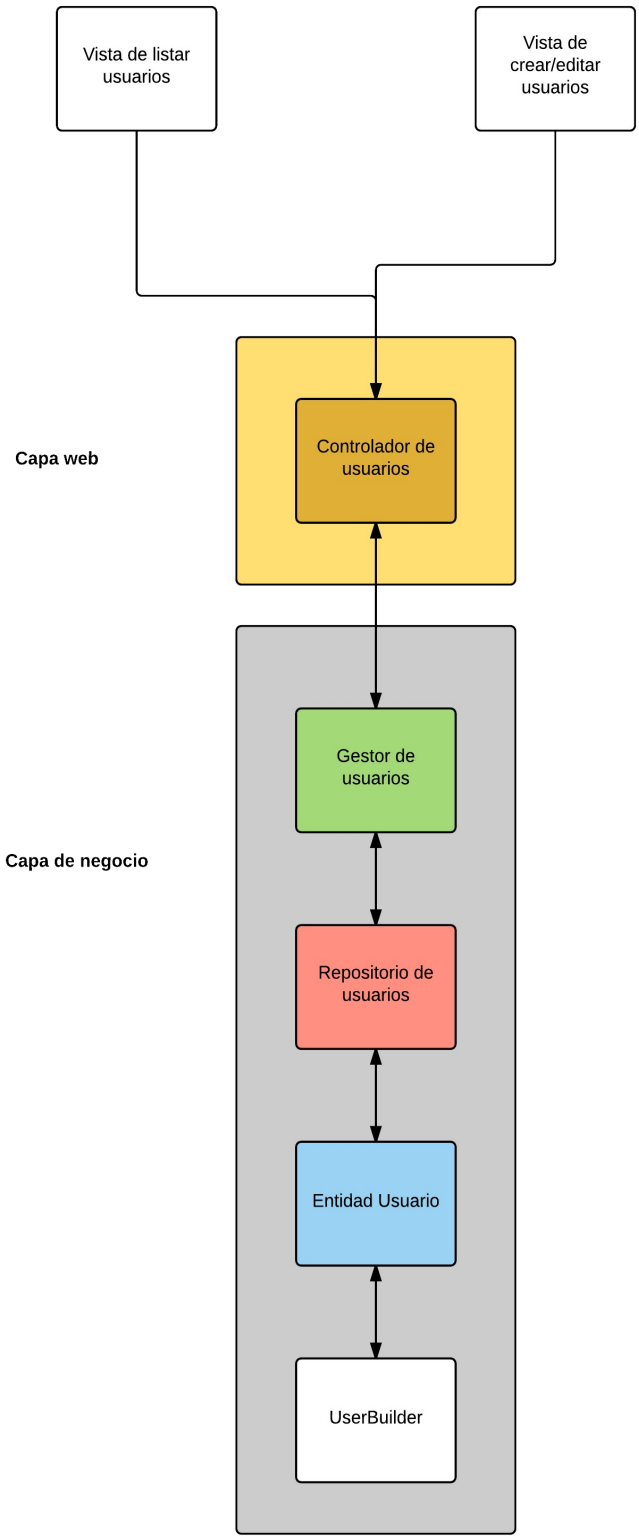

Ilustración 33: Arquitectura del módulo de gestión de usuarios

5 Conclusiones y trabajo futuro

5.1 Resumen de contribuciones

El desarrollo de este proyecto de Sistemas Informáticos, me ha supuesto un reto personal teniendo la oportunidad de aprender a utilizar un gran número de tecnologías que son utilizadas en la actualidad en el mundo empresarial y que por tanto he podido y podré aplicar durante mi carrera profesional.

El desarrollo en GitHub ha facilitado que, pese a que el alumno ha liderado el desarrollo de la aplicación, el tutor ha acompañado las explicaciones y los consejos con un *pull request* para solventar alguna duda puntual.

Durante el trabajo de este año se ha logrado desarrollar un prototipo de portal para la Facultad de Informática que utiliza un diseño que facilita su uso en múltiples dispositivos, en particular, en los dispositivos móviles. Asimismo, la elección del framework Spring como base para construir la aplicación ha permitido modularizar la aplicación, de modo que ha sido posible añadir módulos nuevos paulatinamente durante el desarrollo del trabajo, llegándose a desarrollar cinco módulos o herramientas, en particular los módulos de gestión de reservas de espacios y solicitud y gestión de tutorías son dos módulos que en la actualidad no existen en el portal interno de la Facultad.

Cabe destacar que el código generado durante este trabajo se ha liberado con una licencia GNU Affero General Public License 3.0, más conocida por **Affero GPL 3.0**. Se ha elegido esta licencia con objeto de permitir el desarrollo y evolución del presente trabajo en otros proyectos de Sistemas Informáticos o trabajos de fin de Grado..

5.2 Trabajo Futuro

Las ideas iniciales eran muchas, pero al ir desarrollando la aplicación, se ha constatado que muchas no eran realizables, por falta de tiempo, al estar el proyecto realizado por un único alumno, en lugar de por varios alumnos, como suele ser habitual.

Por otra parte, se ha constatado que una aplicación para gestionar información de la facultad es una idea útil y factible, que en un futuro podría modernizar un poco la burocracia y la transmisión de información, ahorrando así tiempo y recursos (escasos

siempre, acentuados en tiempos de recortes).

Por falta de tiempo, no se han podido materializar ciertas propuestas.

- Implementación del procedimiento de pre-reserva de la asignatura “Sistemas Informáticos”.

- Implementación del procedimiento de registro y asistencia a los ciclos de conferencias.

- Integración del login utilizando las cuentas de Google, de modo que no fuera necesario tener que almacenar las contraseñas de los usuarios.

- División de los módulos en proyectos / aplicaciones independientes, que pemitirían depurar, probar y mantener los módules de manera sencilla.

- Terminar de almacenar el texto estático en archivos de configuración que permitan su fácil traducción y que posibilite el poder configurar con múltiples idiomas el portal.

6 Guía de instalación

Debido a la naturaleza open source del proyecto, cualquier persona puede copiar el código y/o modificarlo para reutilizar la aplicación. Por ello, se pretenden dar todas las facilidades para que los usuarios potenciales de la puedan acceder y configurar la aplicación.

Siendo Github un portal de espíritu colaborativo, y la aplicación software libre, cualquiera puede hacer un “fork” de la aplicación, es decir, crear un repositorio con base a la aplicación de gestión de la fdi, y hacer modificaciones, pudiendo hacer un “pull-request” para que los autores puedan incluir las modificaciones hechas por terceras personas en el código original.

6.1 Acceso al código fuente

Para descargar el código de la aplicación basta con usar git u otro sistema compatible, y acceder con dicha aplicación a la URL de github **https://github.com/patsancu/app-fdi.** Si no se dispone de una aplicación que pueda descargar el código, se puede descargar comprimida en formato zip desde la URL **https://github.com/patsancu/app-fdi/archive/master.zip** .

6.2 Configuración

6.2.1 Datos de prueba

Se han incluido varios scripts sql de carga de datos de prueba: users-init.sql, avisos-init.sql, espacios-init.sql, tutorias-init.sql y users-init.sql que se pueden modificar (prestando atención a las restricciones de Hibernate), sin causar problemas.

6.2.2 Archivo de configuración de propiedades de Maven

Por comodidad en el desarrollo, se ha elegido configurar las propiedades de Maven en las el archivo del usuario, aunque también se podría haber configurado en el archivo de configuración global de Maven.

En dicho archivo (en Linux guardado en /home/nombreUsuario/.m2/settings.xml, consultar [30] para otros sistemas), se deberán establecer ciertas propiedades, que no deberían incluirse nunca con el resto del código, debido a su carácter sensible.

6.2.2.1 Claves de twitter

Para obtener las claves que usará la aplicación para interactuar con la API de Twitter, deberá accederse a https://apps.twitter.com/ e identificarse con una cuenta de Twitter. Se llegará a la pantalla mostrada en la siguiente imagen.

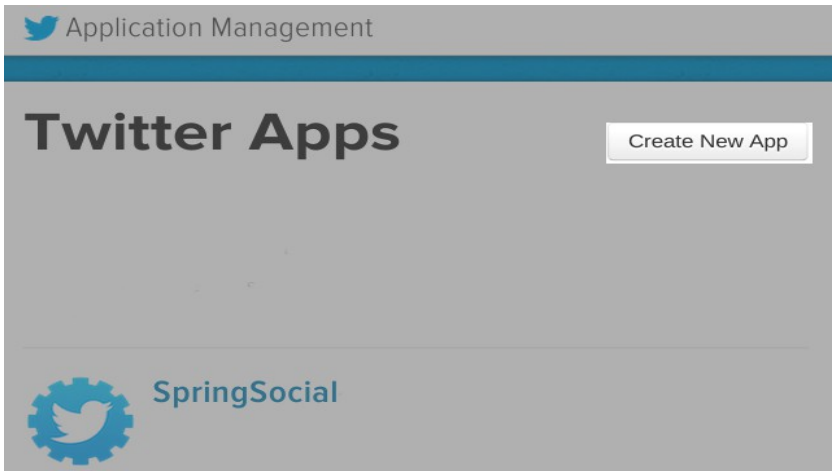

Ilustración 34: Creación de Apps con Twitter

Una vez en esta pantalla, se pulsa sobre “Create New App”, y se llegará a la siguiente pantalla de configuración de la aplicación.

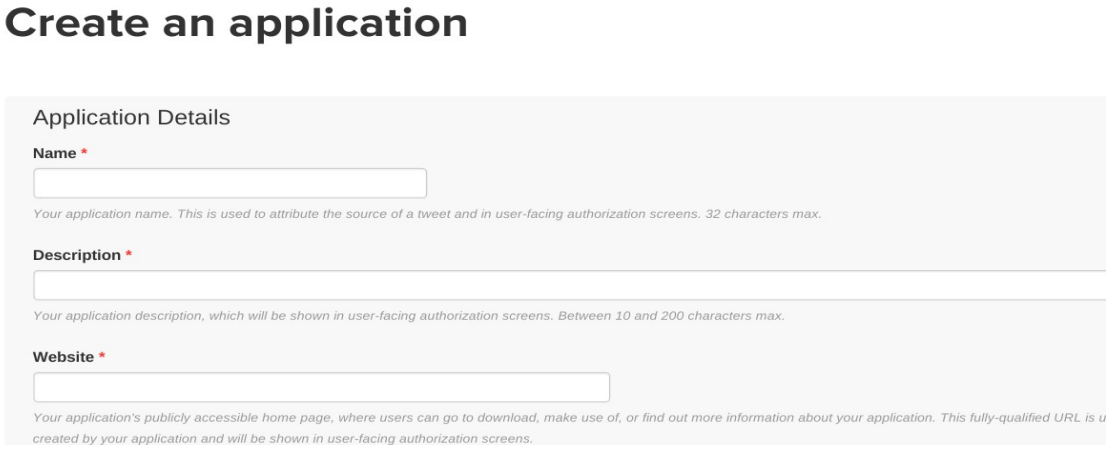

Ilustración 35: Introducción de datos para la app

Después de rellenar correctamente los datos requeridos, se habrá creado la aplicación con Twitter, y se podrán ver los valores de las claves necesarias para utilizar la opción de tuitear avisos. En el archivo “settings.xml” elegido, deberán indicarse: la clave de

consumidor <twitter.consumerKey>, el secreto <twitter.consumerSecret>, el token de acceso <twitter.accessToken> y el secreto del token <twitter.accessTokenSeceret>.

6.2.2.2 Otras claves

Para que la funcione la opción de remember me (para que el sistema recuerde al usuario) de Spring Security, debe añadirse una clave, como esta:

<fdi.remember.me.key>RellenarClave</ fdi.remember.me.key>

6.2.2.3 Datos de usuarios

Para incluir datos de usuarios, con el fin de probar en profundidad la funcionalidad de las tutorías, se han incluido ciertos datos sensibles, como cuentas de correo, en el archivo “settings.xml”. Si se quieren configurar más o menos usuarios, no hay más que añadirlos según el formato establecido en el archivo incluido “users-init.sql”.

6.2.2.4 Base de datos

En esta aplicación se usa la base de datos HSQLDB, pero se puede cambiar fácilmente. Para cambiar la base de datos usada, o cualquiera de los parámetros de acceso a ella, deberán realizarse los siguientes pasos:

- En el fichero de configuración de proyecto de Maven, “pom.xml”, deben cambiarse los valores de las propiedades del conector **jdbc**: <jdbc.driverClassName>, <jdbc.url>, <jdbc.url>, <jdbc.password>, los valores de **hibernate** <hibernate.dialect>, <hibernate.hbm2ddl.auto> y <hibernate.show_sql>.

- Añadir al “pom.xml” las dependencias requeridas para el uso de la base de datos deseada.

- Asegurarse de que no hay incompatibilidades con Hibernate al cambiar de base de datos.

6.3 Ejecución

Para ejecutar el proyecto debe instalarse un servidor **Apache Tomcat**, para ejecutar los servlets del proyecto, y **Maven,** para construirlo.

6.3.1 Ejecución desde IDE

Para realizar pruebas, se recomienda instalar el IDE STS, disponible en https://spring.io/tools. También puede usarse el IDE Eclipse.

6.3.2 Ejecución directa

Para ejecutar el proyecto sin IDE, hay que construir la aplicación ejecutando la instrucción **mvn install**, lo que creará un archivo war. Este archivo debe copiarse a la ubicación correcta de tomcat (normalmente en el directorio $CATALINA_BASE/webapps), o desplegarse a través de la interfaz gráfica incluida con Tomcat.

7 Bibliografía

Bibliografía

1: Murtagh, Rebecca, ** ,
http://searchenginewatch.com/sew/opinion/2353616/mobile-now-exceeds-pc-the-biggest-shift-since-the-internet-began
2: Perez, Sarah, ** ,
http://techcrunch.com/2014/08/21/majority-of-digital-media-consumption-now-takes-place-in-mobile-apps/
3: Knight, Kayla, **,
http://www.smashingmagazine.com/2011/01/12/guidelines-for-responsive-web-design/
4: Fielding, Jonathan, Beginning Responsive Web Design with HTML5 and CSS3, 2014
5: UCM, ** , http://www.ucm.es/faq/gestor-web/
6: Makino, Takaki, **,
http://googlewebmastercentral.blogspot.de/2015/04/rolling-out-mobile-friendly-update.html
7: Mike Cohn, User Stories Applied: For Agile Software Development, 2004
8: Ambler, Scott, ** , http://www.agilemodeling.com/artifacts/userStory.htm
9: Cohn, Mike, ** ,
https://www.mountaingoatsoftware.com/agile/scrum/product-owner
10: Eckel, Bruce, Thinking in Java, 2007
11: Crockford, Douglas , JavaScript: The Good Parts , 2008
12: Maven, ** , https://maven.apache.org/what-is-maven.html
13: Pivotal, Pro Spring, 2014.
14: Spring, ** ,
http://docs.spring.io/spring-framework/docs/current/spring-framework-reference/html/overview.html#overview-modules
15: Wikipedia, ** , https://en.wikipedia.org/wiki/Aspect-oriented_programming
16: Spring, ** , http://projects.spring.io/spring-security/
17: Spring, **,
http://docs.spring.io/spring-social/docs/1.0.3.RELEASE/reference/html/overview.html
18: Spring, ** , http://projects.spring.io/spring-data-jpa/
19: Twitter, ** , http://getbootstrap.com/
20: Hibernate, ** , http://hibernate.org/orm/
21: Hibernate, ** , http://hibernate.org/orm/what-is-an-orm/
22: Eclipse, **,
http://epf.eclipse.org/wikis/openup/core.tech.common.extend_supp/guidances/guidelines/entity_control_boundary_pattern_C4047897.html
23: Fowler, Martin, Patterns of Enterprise Application Architecture, 2002
24: Shvets, Alexander, ** , https://sourcemaking.com/design_patterns/builder
25: Wikipedia, **, https://en.wikipedia.org/wiki/Representational_state_transfer
26: HSQL, **, http://hsqldb.org/web/hsqlFeatures.html
27: Wang, Thomas, ** , https://gist.github.com/badboy/6267743

28: Crockford, Douglas, **, http://crockford.com/wrmg/base32.html
29: Defuse Security, ** , http://crackstation.net/hashing-security.htm
30: Maven, **, https://maven.apache.org/settings.html

** : accedido por última vez el 5 de junio de 2015.